KDnet-RUL: A Knowledge Distillation Framework to Compress Deep Neural Networks for Machine Remaining Useful Life Prediction

Qing Xu[1], Zhenghua Chen[1]*, Keyu Wu[1], Chao Wang[2], Min Wu[1], and Xiaoli Li[1]

***Abstract*—Machine remaining useful life (RUL) prediction is vital in improving the reliability of industrial systems and reducing maintenance cost. Recently, long short-term memory (LSTM)-based algorithms have achieved state-of-the-art performance for RUL prediction, due to their strong capability of modeling sequential sensory data. In many cases, the RUL prediction algorithms are required to be deployed on edge devices to support real-time decision making, reduce the data communication cost and preserve the data privacy. However, the powerful LSTM-based methods which have high complexity cannot be deployed to edge devices with limited computational power and memory. To solve this problem, we propose a knowledge distillation framework, entitled KDnet-RUL, to compress a complex LSTM-based method for RUL prediction. Specifically, it includes a generative adversarial network based knowledge distillation (GAN-KD) for disparate architecture knowledge transfer, a learning-during-teaching based knowledge distillation (LDT-KD) for identical architecture knowledge transfer and a sequential distillation upon LDT-KD for complicated datasets. We leverage simple and complicated datasets to verify the effectiveness of the proposed KDnet-RUL. The results demonstrate that the proposed method significantly outperforms state-of-the-art KD methods. The compressed model with 12.8 times less weights and 46.2 times less total float point operations even achieves a comparable performance with the complex LSTM model for RUL prediction.**

***Index Terms*—Knowledge distillation, model compression, generative adversarial network, remaining useful life prediction.**

I. Introduction

Machine remaining useful life (RUL) prediction is of great importance for real industry [1], [2], [3], [4], [5]. It is able to reduce the maintenance cost and improve the reliability of industrial systems. However, accurate prediction of machine RUL is still challenging, due to the high complexity of modern industrial systems. To predict machine RUL, many advanced methods have been developed. Generally, they can be divided into two different categories, i.e., model-based and data-driven. Model-based solutions intend to explicitly model the relationship between sensory data and RUL [6], [7]. Since the industrial systems become more and more complex, the explicit modeling is extremely difficult. Alternatively, data-driven solutions aim to learn the relationship directly from data without knowing the physical model of a system [8], [9]. They become very promising techniques for RUL prediction, especially for complicated industrial systems.

Conventional machine learning algorithms are widely used data-driven methods to predict machine RUL [10], [8]. For conventional machine learning-based RUL prediction, the first step is to perform feature engineering which manually extracts representative features from the sensory data based on expert knowledge. Then, machine leaning algorithms, such as support vector regression, decision tree, random forest, etc., can be adopted to predict RUL based on the extracted features. However, conventional machine learning-based RUL prediction requires to extract features based on domain knowledge which may not be available all the time. Besides, the feature extraction and RUL prediction cannot be jointly optimized in conventional machine learning methods, which also hinders their performance.

Recently, deep learning has achieved great successes in many challenging domains, including RUL prediction [11], [12]. The greatest merit of deep learning is that it is able to automatically learn representative features from data without human intervention and perform RUL prediction simultaneously, leading to a superior performance. One of the most popular deep learning algorithms is convolutional neural network (CNN) which has achieved remarkable performance for image classification [13]. Due to the unique structure of CNN, it is very efficient for feature learning and can be trained in parallel. It has also been utilized for RUL prediction and outperformed conventional machine leaning algorithms [14], [15]. Another popular deep learning algorithm for RUL prediction is long short-term memory (LSTM) which is specifically designed for analyzing sequential data with temporal information [16], [17]. Since the sensory data for machine RUL prediction are typical time series with temporal information, the LSTM network is naturally suitable for RUL prediction. Recent studies [16], [17], [18] have shown that the LSTM outperforms the CNN for RUL prediction. However, the LSTM generally has much higher computational complexity than CNN due to its unique structure of cascade connection.

This work is supported by the A*STAR Industrial Internet of Things Research Program under the RIE2020 IAF-PP Grant A1788a0023, and partially supported by the National Key Research and Development Program of China (under Grant 2017YFA0700900, 2017YFA0700903), and National Science Foundation of China (No. 61976200). (Zhenghua Chen is the corresponding authors.)

[1] Qing Xu, Zhenghua Chen, Keyu Wu, Min Wu and Xiaoli Li are with Institute for Infocomm Research, A*STAR, Sinagpore (Email: chen0832@e.ntu.edu.sg, {xu_qing,wu_keyu,wumin,xlli}@i2r.a-star.edu.sg).

[2] Chao Wang is with School of Computer Science, University of Science and Technology of China, Hefei, China (Email: cswang@ustc.edu.cn).

In many real-world scenarios, the RUL prediction algorithms need to be deployed on edge devices, which have limited computational resources and memory, for timely response and security concerns. Thus, the industry generally prefers a learning algorithm which can achieve accurate RUL prediction and is also very efficient (e.g., small size and fast inference). The current deep learning algorithms are either too complicated or with limited performance.

To deal with these issues, model compression techniques have been proposed to compress deep neural networks for edge deployment. For instance, parameter quantization methods [19], [20] compress the original network by using less bits to represent the weights. They can achieve significant speed-up but also result in accuracy loss [21]. Another commonly used method for model compression is weight pruning [22], which aims to remove unnecessary parameters in a trained deep neural network. Although the weight pruning is able to reduce model storage size, it cannot improve the efficiency in terms of training or inference time. Other methods like matrix decomposition [23], [24] have also shown the capability of reducing model size, but they only addresses the storage complexity issue of deep models and have similar drawbacks as the weight pruning method. Relatively, knowledge distillation has shown great promise in reducing not only model storage size but also model efficiency [25], [26].

In this paper, we propose a novel knowledge distillation framework, entitled KDnet-RUL, to compress deep learning models for RUL prediction. Specifically, we firstly design a generative adversarial network based knowledge Distillation (GAN-KD) for disparate architecture knowledge transfer, which distills the knowledge from a powerful and complicated LSTM model to a simple CNN model. Then, a learning-during-teaching knowledge distillation (LDT-KD) for identical architecture knowledge transfer is proposed to enhance the performance of the CNN model learned by GAN-KD. For complicated RUL prediction scenarios, e.g., data with multiple operation conditions, we leverage a sequential distillation scheme upon the LDT-KD for accurate and robust RUL prediction. The performance of the proposed KDnet-RUL method is evaluated by using both simple and complex datasets.

The main contributions of the proposed method are summarized as follows:

- We propose a knowledge distillation framework, named KDnet-RUL, which distills knowledge from a complicated LSTM model to a simple CNN model for efficient RUL prediction. The efficient CNN model can thus be deployed on resource constrained edge devices.
- For knowledge distillation between disparate architectures, i.e., from LSTM to CNN, a GAN-KD method is proposed. It attempts to minimize the discrepancy between the features learned from LSTM and CNN by using a GAN technique.
- To enhance the performance of CNN, we propose a LDT-KD method for knowledge distillation between identical architectures.
- In complicated scenarios where multiple working conditions are involved for RUL prediction, we propose a sequential distillation scheme upon LDT-KD to further enhance the performance of the learned CNN model.

The rest of the paper is organized as follows: Section II reviews some related works on RUL prediction and KD. Section III presents the deep neural networks for RUL prediction, followed by the disparate and identical architecture knowledge transfer. Section IV firstly describes the data for evaluation and the experimental setup. Then, the experimental results, ablation study and sensitivity analysis are introduced. Section V concludes this paper and shows some potential future works.

II. Related Work

A. RUL Prediction

Deep learning for RUL prediction has gained increasing attention due to its ability of modelling complex machinery degradation process [27]. Various deep learning methods, such as Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM), have been shown to be effective for RUL prediction tasks. Babu et al. proposed a novel CNN-based model to estimate RUL of airplane engines by using sliding windows on the raw sensory data as input samples [14]. Instead of directly feeding the raw sensory data into CNN models, Zhu et al. transformed the sensory data to derive the Time Frequency Representation (TFP) of each sample [15]. Then, a Multiscale Convolutional Neural Network (MSCNN) was developed with these samples for RUL prediction. Even though the CNN-based models have already outperformed traditional methods, such as Multilayer Perceptron (MLP) and Support Vector Machines (SVM), they are not naturally designed for sensory data with temporal information.

To better capture the temporal information of sensory data, Zheng et al. employed a LSTM network to model the long-term dependency characteristic of data for RUL prediction [16]. Hence, such a LSTM method achieved a better performance than traditional machine learning and CNN approaches. Thereafter, several LSTM-based approaches, such as bidirectional LSTM [17] and attention-based LSTM [18], were proposed to further improve RUL prediction accuracy. However, LSTM-based models often have high computational complexity, and thus it is very difficult to deploy them on edge devices with limited computing resources. To address this problem, model compression methods can be adopted for LSTM models to reduce their complexity and preserve the performance as much as possible.

B. Knowledge Distillation

Knowledge Distillation (KD), also known as a teacher-student strategy, is widely applied for model compression. It was firstly introduced by [28], which refers to training a shallow network (i.e., *Student*) by mimicking the output of a larger and deeper network (i.e., *Teacher*). Hinton et al. further generalized it by introducing a temperature variable to soften the logits from the cumbersome model as "soft target" [29].

Subsequently, various methods have been proposed for efficient knowledge transfer between the teacher and student for model compression. To improve the generalization capability

of thin but deep student, Romero et al. introduced a hint-based pre-training strategy to guide the student to learn intermediate feature representations close to the teacher's [30]. The authors in [31] proposed to transfer the attention maps with different levels from a teacher network and showed significant improvements. Tian et al. proposed a contrastive learning approach to force the student to generate close representations as the teacher for the same inputs, while generating distant representations for different inputs [32]. The GAN-based architectures were also adopted to align the source of logits [33] or feature maps [26] for knowledge transfer.

Note that the softened logits output of the teacher in the aforementioned classification tasks can provide additional knowledge about the correlations of class labels [29]. Therefore, most of previous KD studies focus on the classification tasks. In fact, KD is also suitable for regression tasks. Chen et al. introduced a bounded regression loss for knowledge distillation on bounding-box regression problems [34]. Combining with hint-based learning, the proposed distillation framework can significantly improve the accuracy compared to the baselines. The authors in [25] proposed an Attention Imitation Loss (AIL) which intended to use the teacher loss as a confidence score for camera pose regression problem. It allows to attentively learn from the predictions in which the teacher has more confidence.

However, most of previous KD studies focus on transferring knowledge between networks with similar architectures, i.e., the student is a simplified version of teacher with less layers or hidden units. It is not clear whether those KD methods are also suitable for disparate architectures, e.g., a LSTM-based teacher and a CNN-based student. Therefore, to fill this gap, we propose a method named GAN-KD for this scenario. Moreover, due to the inherent difference between LSTM and CNN, we propose a method named LDT-KD to further optimize CNN-based student learned from GAN-KD.

III. Methodology

In this section, we present a framework called KDnet-RUL to transfer knowledge between disparate and identical network architectures for RUL prediction. The overall KDnet-RUL pipeline is depicted as Fig. 1. To be specific, a GAN-KD approach is proposed to transfer knowledge between different network structures, i.e., from LSTM to CNN. A LDT-KD approach is proposed to transfer knowledge between identical network structures, i.e., from CNN to CNN. Moreover, a sequential self-distillation scheme upon LDT-KD is designed to further improve the performance of RUL prediction on complex datasets with multiple operating conditions.

A. Deep Neural Networks for RUL Prediction

To precisely estimate the RUL for mechanical systems, it is desirable to design deep neural networks (e.g., LSTM or CNN) that are capable of modeling the temporal dependency of multivariate sensory data. Such networks normally consist of a feature extractor and a regression module. In particular, the feature extractor extracts the features from the input sensor data. The extracted feature maps are then fed into the regression module to predict the RUL. The regression module generally contains several fully-connected (FC) layers.

To demonstrate the effectiveness of our proposed pipeline, we first design a LSTM-based network that serves as a powerful but luxurious teacher, considering that it achieves state-of-the-art performance for RUL prediction [16], [17], [18]. Subsequently, a dilated CNN-based network is adopted as the student, which ideally can maintain comparable performance as the teacher but with much less complexity. This dilated CNN-based structure has shown promising capability on handling sequential data [35], and thus we use it as the student network as shown in Fig. 2.

B. Disparate Architecture Knowledge Transfer

As aforementioned, LSTM networks are too complex to be deployed on resource-constrained edge devices. Simple CNN networks are suitable for edge deployment. However,

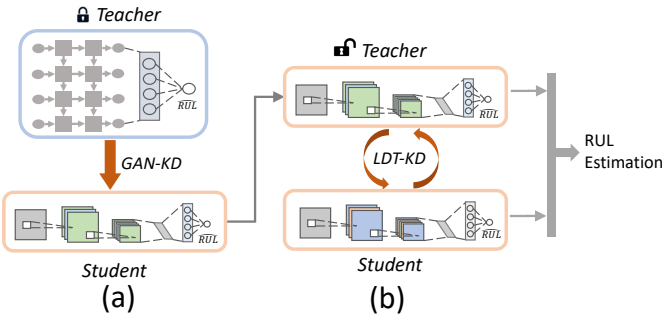

Fig. 1: The proposed KDnet-RUL framework: (a) GAN-KD for disparate network architectures; (b) LDT-KD for identical network architectures.

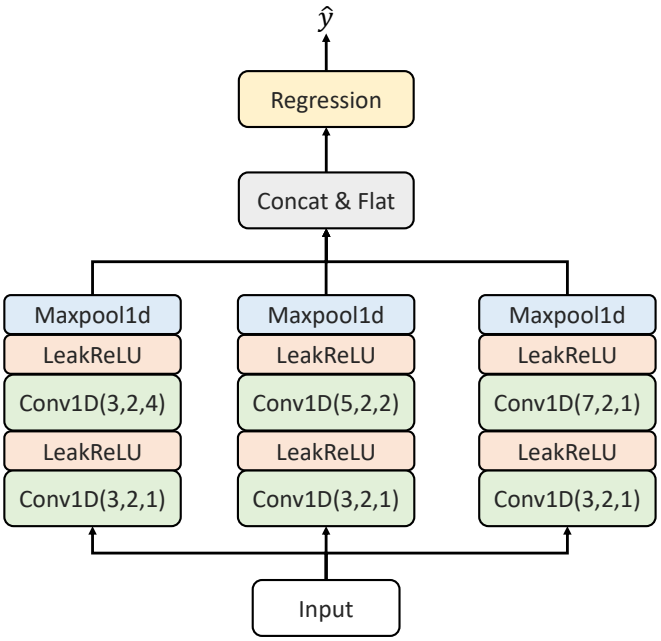

Fig. 2: Dilated CNN-based Student Architecture. Conv1D(3, 2, 1) refers to a 1D convolution layer with kernel size as 3, stride as 2 and dilation as 1.

they are usually not able to achieve desirable performance as LSTM models. To address this dilemma, we firstly propose a GAN-based knowledge distillation method called GAN-KD for knowledge transfer between disparate architectures, i.e., from LSTM to CNN. Particularly, as shown in Fig. 3, we distill the knowledge from a complicated LSTM structure to a simple CNN structure in GAN-KD to improve the performance of the CNN model. In our GAN-KD, both the teacher (LSTM) and student (CNN) consist of a feature extractor and a regression module for RUL prediction. We thus adopt a two-stage training scheme for our GAN-KD. Specifically, we train the feature extractor by feature distillation and the regression module by knowledge distillation separately. Next, we introduce the feature distillation and knowledge distillation in details.

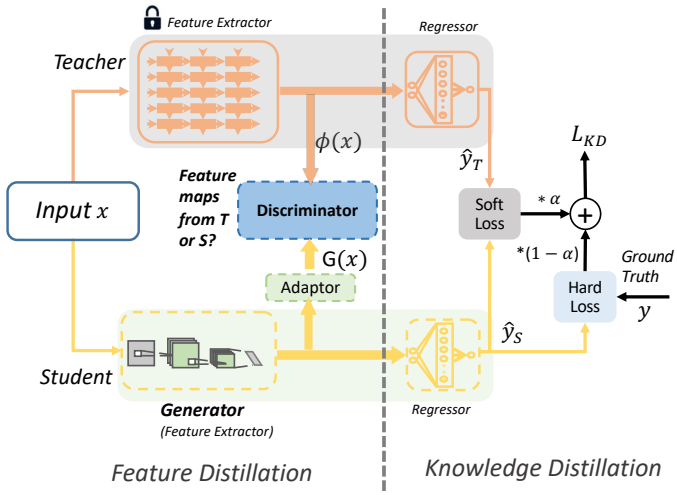

Fig. 3: GAN-KD for disparate network architectures. All network blocks with dash lines are trainable and those with solid lines are locked during training.

1) Feature Distillation: We design a GAN, which contains a Generator (G) and a Discriminator (D), for feature distillation. In particular, the feature extractor of the CNN-based student is considered as the generator G in the GAN as shown in Fig. 3. Meanwhile, the discriminator D is designed to maximize the similarity between the CNN-based and the LSTM-based feature extractors and thus improve the CNN-based feature extractor.

Given that x is the input sensory data, and $x \in \mathbb{R}^{T\times n}$, where T is the window size and n is the number of sensors, $\phi(x)$ is the output of feature extractor in teacher network, while $G(x)$ is the output of feature extractor in student network. The discriminator D, as a binary classification network, aims to identify if the feature map is from the teacher's or student's feature extractor. Here, D and G play a two-player minimax game in which D aims to maximize the probability of correctly classifying $\phi(x)$ and $G(x)$ from teacher and student respectively, and G aims to minimize the probability that D will predict $G(x)$ from student. The objective function can be expressed as follows:

$$\min_G \max_D V(D,G) = \mathbb{E}_x[log(D(\phi(x))) + log(1 - D(G(x)))] \tag{1}$$

At each iteration of training stage, we firstly fix the G and train D by maximizing the following loss function L_D:

$$L_D = log(D(\phi(x))) + log(1 - D(G(x))) \tag{2}$$

Then, we fix D and start to train G by minimizing the probability $log(1 - D(G(x)))$. We further mix this GAN objective with the L_1 distance between student's and teacher's features, denoted as L_G:

$$L_G = log(1 - D(G(x))) + \lambda * \|\phi(x) - G(x)\|_1, \tag{3}$$

where λ is a hyper-parameter to control the contribution of L_1 distance in the final loss L_G. Minimizing L_G can thus help to easily achieve the equilibrium of G generating perfect features as teacher's and D guessing with 50% accuracy.

We alternately repeat the above generator and discriminator training process, i.e., iteratively minimizing L_D and maximizing L_G. Eventually, the student is able to generate the feature maps similar to the teacher's.

2) Knowledge Distillation: Knowledge distillation by logits or soft labels has already been proved to be effective for training the student in classification tasks. In this paper, we deal with the regression task for RUL prediction. Hence, we attempt to utilize predictions from the teacher for knowledge distillation, similar to the logits or soft labels in classification tasks. In particular, we define the *Soft Loss* L_{Soft} as the difference between student's prediction and teacher's prediction in Equation (4). We also have the *Hard Loss* L_{Hard} which is the difference between student's prediction and ground truth (i.e., real labels) in Equation (5). The loss function for knowledge distillation L_{KD} is then defined as the weighted combination of L_{Soft} and L_{Hard} in Equation (6).

$$L_{Soft} = \|\hat{\mathbf{y}}_S - \hat{\mathbf{y}}_T\|_2, \tag{4}$$
$$L_{Hard} = \|\hat{\mathbf{y}}_S - \mathbf{y}\|_2, \tag{5}$$
$$L_{KD} = \alpha * L_{Soft} + (1-\alpha) * L_{Hard}. \tag{6}$$

Here, $\hat{\mathbf{y}}_S$, $\hat{\mathbf{y}}_T$ represent predictions of student and teacher networks, respectively, and $\mathbf{y}$ is the ground truth. α is a hyper-parameter to adjust the weight of hard and soft losses. By minimizing the loss for knowledge distillation L_{KD}, we learn the regressor module in the student for RUL prediction.

C. Identical Architecture Knowledge Transfer

In the above section, GAN-KD can help to learn a simple CNN model by distilling the knowledge from LSTM for RUL prediction. However, the learned CNN may not be optimal in terms of prediction performance, due to the inherent difference between CNN and LSTM. In this section, we aim to further improve the CNN learned by our GAN-KD via knowledge distillation between identical network structures.

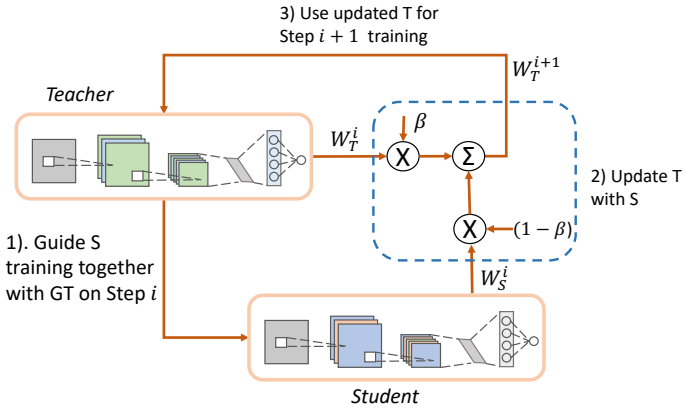

Fig. 4: LDT-KD Architecture. T is the Teacher model, S represents the Student model, and GT represents Ground Truth.

1) Learning-during-teaching: A few previous studies [36], [29] have already proven the feasibility of transferring knowledge between models with identical architectures. Hinton et al. demonstrated the effectiveness of distilling knowledge from an ensemble of models into a single model with the same architecture [29]. However, pre-training a set of models for ensemble is often time-consuming. On the other hand, Yim et al. proposed a Flow of Solution Procedure (FSP) matrix (relationship of outputs from two layers) to transfer the knowledge flow between two identical DNN networks [36]. However, it is not straightforward on how to choose the proper layers to calculate FSP.

In this paper, we propose a method called learning-during-teaching knowledge distillation (LDT-KD) to update both the student and teacher in a closed-loop process as shown in Fig. 4. Here, the teacher and student in Fig. 4 have the same network structure and the same set of model weights. The teacher in LDT-KD is directly copied from the student learned by GAN-KD. To accelerate the convergence of the teacher model, we pre-train the student in LDT-KD for several epochs with conventional KD strategy before performing the closed-loop process in Fig. 4. At each training step, we first update the weights of the student with gradient descent under the supervision of ground truth and soft labels from the teacher, i.e., by minimizing the KD loss in Equation (6). Second, we update the weights of the teacher using the exponential moving average of the student weights, inspired by the mean teacher model in [37]. It can be expressed as follows:

$$W_T^{i+1} = \beta * W_T^i + (1-\beta) * W_S^i, \tag{7}$$

where W_T^i and W_S^i represent the weights of the teacher and student at training step i, respectively. β is a smoothing parameter determining how much historical information of the teacher model will be carried forward for the update. Once the teacher weights are updated, we repeat the above two steps until the stopping criteria is satisfied, e.g., the performance of the teacher on the validate data starts to drop.

2) Sequential distillation: Our empirical study shows that the performance of LDT-KD is superior and stable for simple datasets. However, its performance is not consistently good for complex datasets, such as datasets for RUL prediction with multiple operating conditions. To stabilize the model training for RUL prediction, we present a sequential distillation scheme upon the LDT-KD. The sequential distillation was firstly proposed by Furlanello et al. in [38] where they sequentially distilled the knowledge from a teacher with identical structure to a student. And in each generation, a new student is required to be initialized with a different random seed. At the end of the procedure, they employed an ensemble of student models from each generation and achieved a remarkable performance.

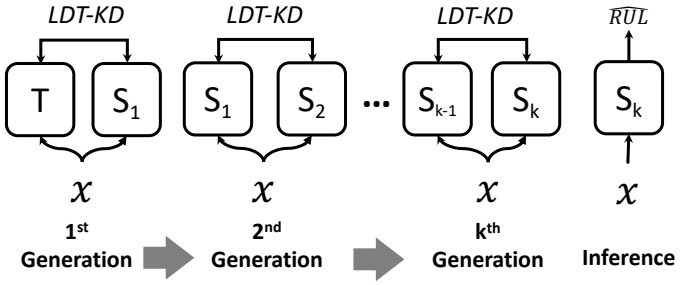

Fig. 5: Sequential Distillation upon LDT-KD

We adopt this sequential training idea upon the LDT-KD module as shown in Fig. 5. However, our method differs from [38] (denoted as BAN) in several aspects. First, the weights of the teacher model are simultaneously updated with these of the student model in our proposed LDT-KD. While the BAN fixes the weights of the teacher model. Second, the BAN applies an ensemble of multiple students from different generations for final prediction. However, this ensemble version is too luxurious for edge devices due to the requirement of more storage memory and longer inference time. For our proposed approach, either the final single student or teacher can be used for RUL prediction. And both teacher and the final student can generalize well. Third, we empirically show that the implementation of sequential training depends on datasets. It is only compulsory to perform sequential distillation for complicated datasets, e.g., datasets with multiple operation conditions.

IV. Experiments

In this section, we evaluate the performance of our proposed KDnet-RUL method to distill the knowledge for RUL prediction.

A. Experimental Data and Setup

1) C-MAPSS Dataset: In our experiments, we used the public C-MAPSS dataset for evaluation, which has been widely used in many previous studies for RUL prediction [14], [18], [17]. This dataset simulates the degradation process of turbofan engines. It consists of four sub-datasets under varying operating conditions and fault modes. For each sub-dataset, it can be further divided into training and testing data, as shown in Table I.

Each trajectory in training and testing data corresponds to an engine and consists of 21 sensor measurements for this engine.

TABLE I: Details of C-MAPSS Dataset

Dataset	Conditions	Fault Modes	Train Traj.	Test Traj.
FD001	1	1	100	100
FD002	6	1	260	259
FD003	1	2	100	100
FD004	6	2	248	249

The training trajectories include all run-to-failure measurements for the engine units, while the testing trajectories only contain the measurements of certain period during degradation. The object is to accurately predict the RUL for the testing engines given their test trajectories.

2) Data Prepossessing: We randomly split each original training data into training and validation by the ratio of 9:1 in terms of engine units. For instance, we randomly select 90 trajectories from the total 100 trajectories in FD001 for model training and the rest 10 trajectories for validation. Then, we applied the following data preprocessing methods to all the training, validation and test data.

First, 7 out of 21 sensors with constant readings (sensor indices 1, 5, 6, 10, 16, 18, and 19) are removed and the rest 14 sensor measurements are utilized to predict the RUL [18], [17]. A min-max normalization is applied to restrict the measurement values within [0,1] to speed up the training process. Particularly, FD002 and FD004 have 6 working conditions and we normalize the data in each working condition for these two datasets. A sliding window with window size T and step size s is adopted to segment the data. For the training and validation data, a sliding window moves with a step size s from the starting cycle to the life-end cycle. For the test data, we extract the last segment with the same window size. As illustrated in Fig. 6, the RUL for the first sample is $L-T$, and the $(i+1)^{th}$ sample has a RUL of $L-T-s*i$, where L is the total engine life cycle.

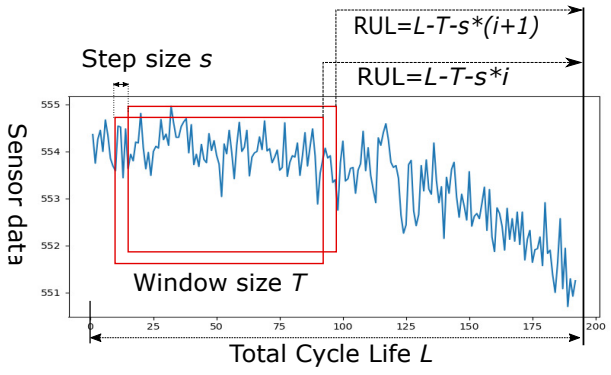

Fig. 6: Data Preprocessing.

In practice, the degradation of system components at the initial stage is not significant and can be negligible. Meanwhile, the system's health degrades along with time when it is getting to the end-of-life. Therefore, we follow the previous studies [18], [39], [40] and apply piece-wise RUL. In particular, if the true RUL is larger than the maximal RUL, it is set to RUL_{max} instead, as shown in Equation 8.

$$\text{RUL} = \begin{cases} L - T - s * i, & \text{if RUL} < \text{RUL}_{max}, \\ \text{RUL}_{max}, & \text{otherwise.} \end{cases} \tag{8}$$

Following the previous studies [18], [39], [40], T, s, and RUL_{max} are set to be 30, 1 and 130 in our experiments, respectively.

3) Experimental Setup: In our experiments, the teacher of our GAN-KD is a 5-layer LSTM model with 32 hidden units in each layer as feature extractor and 2 FC layers as regression module. After properly training and hyperparameter tuning, we can obtain a decent performance for the teacher on the RUL prediction task. Subsequently, we develop a compact student, which consists of a feature extractor with dilated CNN structure and a regression module with 2 FC layers. We denote the student model, which is firstly trained from scratch under the supervision of ground truth only, as "Student Only". The proposed GAN-KD is evaluated by training a new student under the supervision of both pre-trained LSTM-based teacher and ground truth. We use the validation set to choose the student model and validate its performance on the test set. The selected student is further employed as the teacher at the open-loop pre-train stage of identical architecture knowledge transferring by using the LDT-KD. Similarly, for the sequential self-distillation, we use the teacher selected by the validation set in previous generation as the teacher for next generation.

For the proposed KDnet-RUL, it consists of GAN-KD, LDT-KD and sequential distillation. Some hyper-parameters need to be determined. Specifically, we set the batch size of 256, learning rate of 1e-3, optimizer of Adam, training epochs of 160 for the proposed method. For GAN-KD, we choose $\lambda = 1.0$ for Equation (3). A grid search is adopted to identify the α in Equation(6) from the range $\alpha \in [0.0, 1.0]$ with a step size of 0.1. For LDT-KD, we use $\beta = 0.99$ for the smoothing parameter. Considering the randomness of model initialization, all reported results are the average of 5 repeats.

The FD001 and FD003 are relatively simple datasets with only one working condition. The empirical study shows that the sequential distillation scheme upon LDT-KD is not required on these simple datasets. While for the complicated FD002 and FD004 datasets with multiple working conditions, we adopt the sequential distillation upon LDT-KD to further improve the performance. Specifically, in experiments, we empirically find that three generations are adequate to achieve a satisfactory performance on FD002 and FD004.

4) Evaluation Metrics: Same as previous works, two commonly used metrics are adopted to validate the proposed method, i.e., Root Mean Square Error (RMSE) and Score function. The RMSE is a standard way to measure the error of model predictions, which is defined as follows:

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

where $\hat{y}_i$ and y_i are the predicted RUL and true RUL, respectively. N is the total number of samples. The Score function, defined as Equation (10), was designed to place

TABLE II: Performance comparison among various approaches on four datasets

	RMSE				Score			
Methods	FD001	FD002	FD003	FD004	FD001	FD002	FD003	FD004
Student Only	15.4	17.03	15.68	17.08	446.12	1575.42	605.90	1601.93
Teacher	13.17	14.47	13.57	16.11	276.39	982.53	349.30	1288.88
Standard KD	15.33	16.51	15.49	16.85	401.53	1198.37	675.38	1361.24
L_1-KD	15.12	16.13	17.60	17.12	390.12	1082.43	831.13	1423.88
L_2-KD	15.14	16.21	15.39	16.99	396.78	1111.07	610.84	1369.45
MMD-KD	14.82	16.42	16.88	17.01	381.39	1104.46	778.39	1339.75
CORAL-KD	14.88	16.53	17.53	17.52	386.53	1229.30	884.68	1564.14
BAN	15.13	15.05	14.86	16.15	440.96	1045.45	546.55	1315.21
KDnet-RUL	**13.68**	**14.47**	**12.95**	**15.96**	**362.08**	**929.20**	**327.27**	**1303.19**

more penalization on late predictions than early predictions, as late predictions may lead to more serious catastrophic consequences. Same as RMSE, the lower the Score is, the better performance the model can achieve.

$$Score = \begin{cases} \sum_{i=1}^{N}(e^{-\frac{\hat{y}_i - y_i}{13}} - 1), & \text{if } \hat{y}_i < y_i, \\ \sum_{i=1}^{N}(e^{\frac{\hat{y}_i - y_i}{10}} - 1), & \text{otherwise.} \end{cases} \tag{10}$$

B. Comparison with Benchmark Approaches

To verify the effectiveness of the proposed method, we have compared with some benchmark approaches, including the standard KD [29], L_1-KD [34], L_2-KD [30]. To demonstrate the effectiveness of proposed sequential distillation upon LDT-KD, we also compare it with the BAN in [38] which is also sequentially trained with self-distillation. Particularly, we use 'Student Only' as the teacher in the 1^{st} generation for the BAN. Note that the BAN does not have the ability of model compression. It can only improve model generalization performance at the expense of model complexity in terms of memory and inference time. In experiments, we use the ensemble of multiple students with five generations for the BAN on RUL prediction. Moreover, considering the disparate network architectures between teacher and student, the feature distillation step can also be treated as a domain alignment task which intends to minimize the discrepancy of feature distributions between teacher and student. In order to further validate the effectiveness of the proposed method, we also explore several domain alignment techniques, combined with our framework, such as Maximum Mean Discrepancy (MMD) [41] and Correlation Alignment (CORAL) [42].

Table II shows the evaluation results of different methods on the four sub-datasets. The "Student Only" which implements a dilated CNN performs the worst due to its compact network structure. The teacher model which is built upon the LSTM structure performs much better than the 'Student Only'. All the KD methods improve the performance of the student. This indicates the effectiveness of the KD algorithms for improving the performance of the student network. The BAN can effectively improve the performance of the student model, especially on the complicated scenarios, i.e., FD002 and FD004. Among all the methods, the proposed KDnet-RUL performs the best in terms of both RMSE and Score. Besides, it achieves a comparable performance to the teacher model. In particular, the proposed method outperforms the teacher network on FD002 and FD003 in terms of both RMSE and Score. For FD004, the proposed KDnet-RUL has a superior performance over the teacher model in terms of RMSE.

To verify the effectiveness of dilated CNN as the student, we further compare dilated CNN with conventional CNN [14] under two different scenarios as shown in Table III. In Case I, we train them from scratch for RUL prediction (i.e., Student Only). In Case II, we train them as the students in our KDnet-RUL framework, under the guidance of LSTM-based teacher. Comparing with conventional CNN [14], the dilated CNN is capable of modeling temporal information in time series sensory data, which is vital for RUL prediction. We can observe that dilated CNN performs better than conventional CNN under two different scenarios as shown in Table III. Moreover, our KDnet-RUL can also improve the performance of conventional CNN via knowledge distillation from LSTM, further demonstrating the effectiveness of our proposed knowledge distillation framework.

Table IV compares the complexities of the teacher and student models. Here, we consider the number of weights and total floating-point operations (TFPO) when comparing model complexity. More weights and TFPO refer to a more complex model. Note that, for the proposed KDnet-RUL, the final model is the student network after training. It can be found that the number of weights of the student model is 12.8 times less than the teacher model. During inference, the student model only requires 52,400 TFPO which is 46.2 times more efficient than the teacher model.

In conclusion, the proposed KDnet-RUL can achieve a comparable performance to a very complex LSTM network, but with a much more efficient structure, i.e., 12.8 times less weights and 46.2 times less TFPO.

C. Ablation Study

Recall that our KDnet-RUL consists of three components, namely, GAN-KD, LDT-KD and sequential distillation. In this section, we conduct an ablation study to analyze how each component affects the model performance. In particular, we derive the following model variants for ablation study.

- LDT-KD: the teacher of this variant is a CNN trained from scratch, i.e., "Student Only".
- GAN-KD: the teacher of this variant is a 5-layer LSTM.
- GAN-KD+LDT-KD: LDT-KD in this variant uses the student learned by GAN-KD as its teacher.

Note that, our empirical study shows that the sequential distillation upon LDT-KD does not present further improvement

on the relatively simple datasets, i.e., FD001 and FD003 with one working condition. One possible reason is that there is a performance threshold for student due to the limited model capability and such threshold can be easily reached for those relatively simple datasets. Therefore, we don't perform the sequential distillation upon LDT-KD on FD001 and FD003, i.e., KDnet-RUL refers to GAN-KD+LDT-KD on FD001 and FD003. For the complicated datasets, i.e., FD002 and FD004 with six working conditions, the sequential distillation upon LDT-KD can further enhance the performance of the KDnet-RUL model. KDnet-RUL refers to the combination of three components on FD002 and FD004.

Fig. 7 illustrates the experimental results of the ablation study on FD001 and FD003. It is clear that both the LDT-KD and GAN-KD can significantly improve the performance of the student network. The KDnet-RUL with both GAN-KD and LDT-KD is able to further enhance the performance, especially on the FD003 dataset, in terms of both RMSE and Score. It even outperforms the powerful and complex teacher network on FD003. This indicates that the proposed GAN-KD and LDT-KD are effective for RUL prediction on simple datasets.

The experimental results on the complicated FD002 and FD004 are shown in Fig. 8. In this scenario, the KDnet-RUL is the combination of all the three components. Similarly, the performance of the student network can be enhanced by the proposed GAN-KD and LDT-KD, except for the RMSE on FD004. When combining the GAN-KD and LDT-KD (i.e., GAN-KD+LDT-KD in 8), the performance improvement is marginal. For the Score values on FD004, including LDT-KD even degrades the performance of the model. However, when combining sequential distillation upon LDT-KD (i.e., with multiple generations of LDT-KD), the performance of model is consistently improved. With several generations (i.e., three generations of LDT-KD in this paper), the performance of the model becomes stable and even better than the teacher network in terms of the RMSE on FD004 and the Score on FD002.

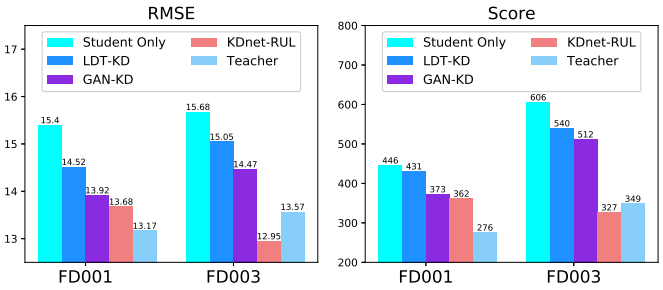

Fig. 7: The results of the ablation study on FD001 and FD003.

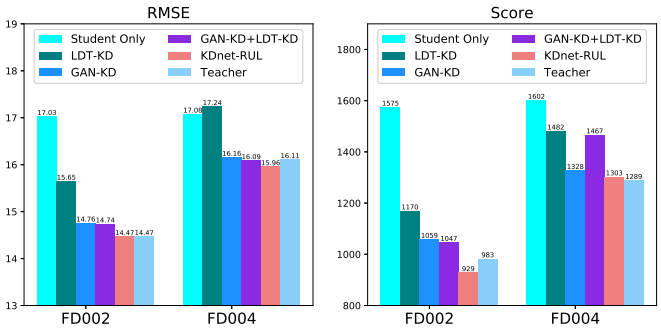

Fig. 8: The results of the ablation study on FD002 and FD004.

In a word, the proposed KDnet-RUL does not require sequential distillation upon LDT-KD (i.e., only one LDT-KD) to ensure a superior performance for simple datasets. While the sequential distillation upon LDT-KD (i.e., several generations of LDT-KD) is compulsory for complex datasets.

D. Parameter Sensitivity Analysis

In this subsection, we conduct the sensitivity analysis for parameters α, λ and β. In particular, we adopt the grid search method on the validation set for parameter tuning.

a) Parameter α: Fig. 9 shows the impact of a key hyper-parameter α in Equation (6), which controls the contribution of truth labels and soft labels when supervising the training of the student network. Two special cases are $\alpha = 0.0$ and $\alpha = 1.0$, which respectively means only using the ground truth and only using the soft labels for training. However, $\alpha = 1.0$ will gradually mislead the student in LDT-KD and Sequential distillation process, such that the performance of teacher and student will degrade rapidly over generations due to the error accumulation if there is no correction from truth labels. Hence, we omit the result of $\alpha = 1.0$ in Fig. 9. As we can see, the soft labels produced by teacher are more contributive to the performance than the ground truth. In most cases, a higher α value tends to yield better performance. In our experiments, we set α as 0.7 on FD001, FD002 and FD003, and 0.8 on FD004. Empirically, we would recommend $\alpha = 0.7$ for our proposed model.

b) Parameter λ: Fig. 10 illustrates the impact of hyper-parameter λ in Eq. (3) on model performance for the four sub-datasets. As we can see, $\lambda = 0.0$ performs worst in terms of RMSE and Score. It demonstrates that integrating L_1 distance during training the generator G can help to improve model performance as aforementioned. With λ increasing from 1 to 10, model gradually performs worse since the training of generator relies more on L_1 distance, which is harmful for disparate architecture knowledge transfer.

TABLE III: Performance Comparison between Conventional CNN and Dilated CNN

Scenarios		RMSE				Score			
		FD001	FD002	FD003	FD004	FD001	FD002	FD003	FD004
Case I: Student Only	Conventional-CNN	17.24	22.79	20.23	23.17	638.92	1481.48	1128.55	2387.43
	Dilated-CNN	15.4	17.03	15.68	17.08	446.12	1575.42	605.90	1601.93
Case II: KDnet-RUL	Conventional-CNN	16.15	19.65	18.22	20.96	491.94	1209.54	797.96	1968.64
	Dilated-CNN	13.68	14.47	12.95	15.96	362.08	929.20	327.27	1303.19

TABLE IV: Comparison of model complexity.

	Teacher (LSTM-based)	Student (CNN-based)	Rate
No. of weights	114,753	8,949	12.8×
No. of TFPO	2,423,360	52,400	46.2×

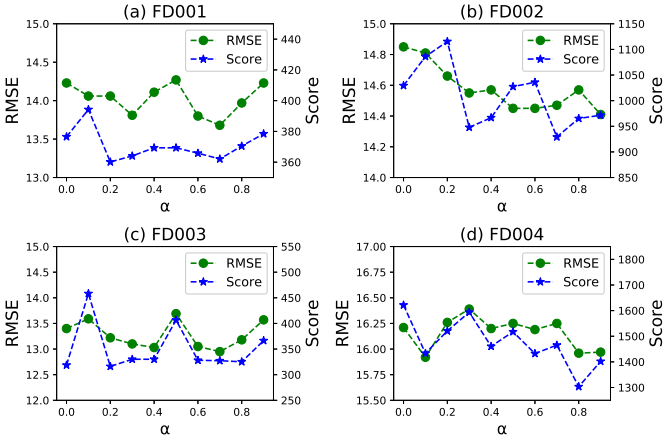

Fig. 9: Sensitivity analysis of parameter α.

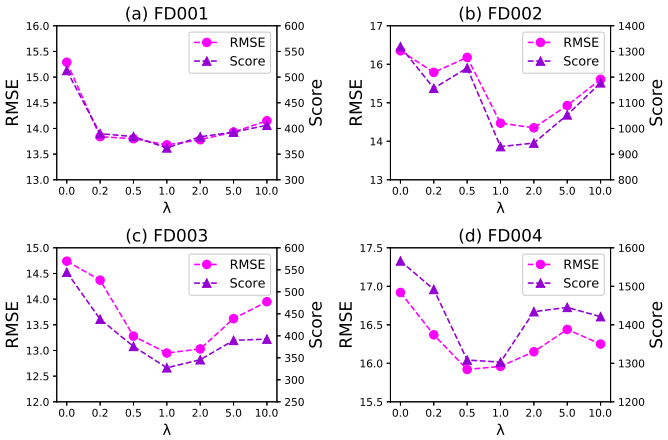

Fig. 10: Sensitivity analysis of parameter λ.

c) Parameter β: Fig. 11 shows the performance of the teacher learned by LDT-KD with different values for the hyper-parameter β in Eq. (7). Small β will dramatically update teacher's model parameters and will make the model hard to converge. Empirically, we would recommend $\beta = 0.99$ for our proposed model.

E. Results on PHM2008 Challenge Dataset

PHM2008 Challenge dataset was used for prognostics challenge competition at the International Conference on Prognostics and Health Management (PHM2008) [4]. It is also widely used to evaluate the performance of models for RUL prediction. Since the true RUL values of the challenge dataset are not released, the results need to be uploaded to NASA Data Repository website, where the RUL Score will be generated for performance evaluation.

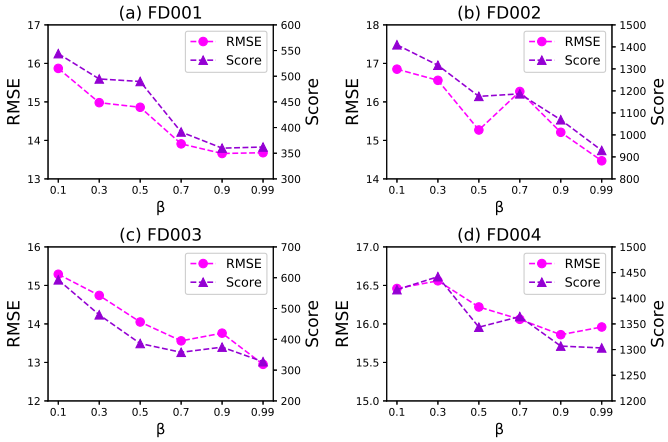

Fig. 11: Sensitivity analysis of parameter β.

TABLE V: Results on PHM 2008 Challenge Dataset

Methods	Score
MLP [14]	3212
SVR [14]	15886
RVR [14]	8242
CNN [14]	2056
LSTM [16]	1862
Attention-LSTM [18]	1584
Teacher	1602
Student Only	3136
Standard KD	2124
L_1-KD	1766
L_2-KD	1704
MMD-KD	1580
CORAL-KD	1962
KDnet-RUL	**1489**

Table V shows the results on the PHM2008 Challenge Dataset. In addition to various KD baselines, we also included the results of various RUL prediction methods (e.g., CNN [14], LSTM [16] and Attention-based LSTM [18]) in Table V. It is clear that various KD methods can effectively help to improve the performance of a compact student ("Student Only"). The proposed KDnet-RUL has a superior performance over not only state-of-the-art RUL prediction approaches but also various benchmark KD methods.

V. Conclusion

In this paper, we proposed a deep model compression framework based on knowledge distillation (KD), named KDnet-RUL, for machine remaining useful life prediction. The KDnet-RUL consists of three components, i.e., generative adversarial network based KD (GAN-KD), learning-during-teaching based KD (LDT-KD) and sequential distillation. A complicated LSTM based model was adopted as a powerful teacher and a dilated convolutional neural network (CNN) was utilized as an efficient student. By using the proposed KDnet-RUL, the student network can achieve comparable performance with the teacher network, but with 12.8 times less weights and 46.2 times less total float point operations.

In the future, we will consider a more real and challenging scenario where the data for training and testing may come from

different distributions, caused by the changing environments or varying machines. For instance, we have built a model based on the collected dataset A from machine A. When directly using this RUL prediction model on a new Machine B which may have different data distributions due to different operation conditions, the performance of the model will significantly degrade. While collecting a new dataset from machine B to re-train the model is tedious and requires lots of efforts. Instead of doing that, we intend to transfer the knowledge learned from dataset A to the new machine B without collecting labeled data from machine B, which is also known as domain adaptation [43], [44]. In this practical scenario, both model compression and domain adaptation need to be considered.

REFERENCES

[1] F. Yang, M. S. Habibullah, T. Zhang, Z. Xu, P. Lim, and S. Nadarajan, "Health index-based prognostics for remaining useful life predictions in electrical machines," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 4, pp. 2633–2644, 2016.

[2] J. Sikorska, M. Hodkiewicz, and L. Ma, "Prognostic modelling options for remaining useful life estimation by industry," *Mechanical systems and signal processing*, vol. 25, no. 5, pp. 1803–1836, 2011.

[3] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation–a review on the statistical data driven approaches," *European journal of operational research*, vol. 213, no. 1, pp. 1–14, 2011.

[4] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–9.

[5] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–6.

[6] R. K. Singleton, E. G. Strangas, and S. Aviyente, "Extended kalman filtering for remaining-useful-life estimation of bearings," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 3, pp. 1781–1790, 2014.

[7] N. Li, Y. Lei, L. Guo, T. Yan, and J. Lin, "Remaining useful life prediction based on a general expression of stochastic process models," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 7, pp. 5709–5718, 2017.

[8] R. Khelif, B. Chebel-Morello, S. Malinowski, E. Laajili, F. Fnaiech, and N. Zerhouni, "Direct remaining useful life estimation based on support vector regression," *IEEE Transactions on industrial electronics*, vol. 64, no. 3, pp. 2276–2285, 2016.

[9] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, "Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2416–2425, 2018.

[10] Z. Tian, "An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring," *Journal of Intelligent Manufacturing*, vol. 23, no. 2, pp. 227–237, 2012.

[11] M. Xia, T. Li, T. Shu, J. Wan, C. W. De Silva, and Z. Wang, "A two-stage approach for the remaining useful life prediction of bearings using deep neural networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3703–3711, 2018.

[12] B. Yang, R. Liu, and E. Zio, "Remaining useful life prediction based on a double-convolutional neural network architecture," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9521–9530, 2019.

[13] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[14] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *International conference on database systems for advanced applications*. Springer, 2016, pp. 214–228.

[15] J. Zhu, N. Chen, and W. Peng, "Estimation of bearing remaining useful life based on multiscale convolutional neural network," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 4, pp. 3208–3216, 2018.

[16] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *2017 IEEE international conference on prognostics and health management (ICPHM)*. IEEE, 2017, pp. 88–95.

[17] C.-G. Huang, H.-Z. Huang, and Y.-F. Li, "A bidirectional lstm prognostics method under multiple operational conditions," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 11, pp. 8792–8802, 2019.

[18] Z. Chen, M. Wu, R. Zhao, F. Guretno, R. Yan, and X. Li, "Machine remaining useful life prediction via an attention based deep learning approach," *IEEE Transactions on Industrial Electronics*, pp. 1–1, 2020.

[19] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *ICLR*, 2014.

[20] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4820–4828.

[21] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *arXiv preprint arXiv:1710.09282*, 2017.

[22] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[23] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. De Freitas, "Predicting parameters in deep learning," in *Advances in neural information processing systems*, 2013, pp. 2148–2156.

[24] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Advances in neural information processing systems*, 2014, pp. 1269–1277.

[25] M. R. U. Saputra, P. P. de Gusmao, Y. Almalioglu, A. Markham, and N. Trigoni, "Distilling knowledge from a deep pose regressor network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 263–272.

[26] W.-C. Chen, C.-C. Chang, and C.-R. Lee, "Knowledge distillation with feature maps for image classification," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 200–215.

[27] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, "Machinery health prognostics: A systematic review from data acquisition to rul prediction," *Mechanical Systems and Signal Processing*, vol. 104, pp. 799–834, 2018.

[28] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

[29] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *NIPS Deep Learning and Representation Learning Workshop*, vol. 1050, p. 9, 2015.

[30] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *in International Conference on Learning Representations (ICLR)*, 2015.

[31] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," *5th International Conference on Learning Representations(ICLR)*, 2017.

[32] Y. Tian, D. Krishnan, and P. Isola, "Contrastive representation distillation," *International Conference on Learning Representations*, 2020.

[33] L. Gao, H. Mi, B. Zhu, D. Feng, Y. Li, and Y. Peng, "An adversarial feature distillation method for audio classification," *IEEE Access*, vol. 7, pp. 105 319–105 330, 2019.

[34] G. Chen, W. Choi, X. Yu, T. Han, and M. Chandraker, "Learning efficient object detection models with knowledge distillation," in *Advances in Neural Information Processing Systems*, 2017, pp. 742–751.

[35] Y. Kim, "Convolutional neural networks for sentence classification," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751, 2014.

[36] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4133–4141.

[37] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in neural information processing systems*, 2017, pp. 1195–1204.

[38] T. Furlanello, Z. C. Lipton, M. Tschannen, L. Itti, and A. Anandkumar, "Born again neural networks," *Proceedings of the 35th International Conference on Machine Learning,(ICML)*, vol. 80, pp. 1602–1611, 2018.

[39] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.

[40] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2306–2318, 2016.

[41] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," *International Conference on Machine Learning(ICML)*, pp. 2208–2217, 2017.
[42] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *European conference on computer vision*. Springer, 2016, pp. 443–450.
[43] X. Li, W. Zhang, N.-X. Xu, and Q. Ding, "Deep learning-based machinery fault diagnostics with domain adaptation across sensors at different places," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 8, pp. 6785–6794, 2019.
[44] M. Ragab, Z. Chen, M. Wu, C. S. Foo, K. C. Keong, R. Yan, and X.-L. Li, "Contrastive adversarial domain adaptation for machine remaining useful life prediction," *IEEE Transactions on Industrial Informatics*, 2020.

Qing Xu received the B.Eng. degree in Measuring Control Technology & Instruments from Southeast University, Nanjing, China, in 2010 and the M.Eng. degree in Instrument Science and Technology from Southeast University, Nanjing, China, in 2015. Currently, he is a research engineer at Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore. His research interests include deep learning, transfer learning, model compression and related applications.

Zhenghua Chen received the B.Eng. degree in mechatronics engineering from University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2011, and Ph.D. degree in electrical and electronic engineering from Nanyang Technological University (NTU), Singapore, in 2017. He has been working at NTU as a research fellow. Currently, he is a scientist at Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore. He has won several competitive awards, such as A*STAR Career Development Award, First Runner-Up Award for Grand Challenge at IEEE VCIP 2020, Finalist Academic Paper Award at IEEE ICPHM 2020, etc. He serves as Associate Editor for Elsevier Neurocomputing and Guest Editor for IEEE Transactions on Emerging Topics in Computational Intelligence and Elsevier Neurocomputing. He is currently the Vice Chair of IEEE Sensors Council Singapore Chapter. His research interests include smart sensing, data analytics, machine learning, transfer learning and related applications.

Keyu Wu received the B.Eng degree in Bioengineering from National University of Singapore, Singapore, in 2013 and the Ph.D degree in Electrical and Electronic Engineering from Nanyang Technological University, Singapore, in 2020. She is currently a Scientist with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore. Her research interests include reinforcement learning, transfer learning, unsupervised learning, path planning, and autonomous navigation.

Chao Wang received the BS and PhD degrees from the University of Science and Technology of China, Hefei, China, in 2006 and 2011, respectively, both in computer science. He is currently an associate professor with the University of Science and Technology of China, Hefei, China. He was a visiting scholar with the Electrical and Computer Engineering Department, University of California at Santa Barbara, Santa Barbara, California, from 2015 to 2016. His research interests focus on multicore and reconfigurable computing. He serves as the associate editor of the ACM Transactions on Design Automations for Electronics Systems (ACM TODAES), IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), Microprocessors & Microsystems, and IET Computers & Design Techniques. He is also the publicity chair of HiPEAC 2015 and ISPA 2014. He is a senior member of the ACM, IEEE and CCF.

Min Wu is currently a senior scientist at Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore. He received his Ph.D. degree in Computer Science from Nanyang Technological University (NTU), Singapore, in 2011 and B.S. degree in Computer Science from University of Science and Technology of China (USTC) in 2006. He received the best paper awards in InCoB 2016 and DASFAA 2015. He also won the IJCAI competition on repeated buyers prediction in 2015. His current research interests include machine learning, data mining and bioinformatics.

Xiaoli Li is currently a dept head and principal scientist at the Institute for Infocomm Research, A*STAR, Singapore. He also holds adjunct professor position at Nanyang Technological University. His research interests include data mining, machine learning, AI, and bioinformatics. He has been serving as area chair/senior PC member/workshop chair in leading data mining and AI related conferences (including KDD, ICDM, SDM, PKDD/ECML, WWW, IJCAI, AAAI, ACL and CIKM). Xiaoli has published more than 200 high quality papers and won numerous best paper/benchmark competition awards.