Toward Deep Learning in Remaining Useful Life Estimation

Vinh Nguyen

School of Science

Bachelor's thesis
Espoo 13.09.2020

Thesis supervisor:

Alexander Nikitin

AALTO UNIVERSITY
SCHOOL OF SCIENCE

ABSTRACT OF THE
BACHELOR'S THESIS

Author: Vinh Nguyen

Title: Toward Deep Learning in Remaining Useful Life Estimation

Date: 13.09.2020 Language: English Number of pages: 5+20

Degree programme: Bachelor's Programme in Data Science

Supervisor and advisor: Alexander Nikitin

Unplanned downtime due to equipment wear or system failure often results in severe consequences such as decrease in production outputs, impairment of product quality, and damage to business reputation. Run-till-failure strategy in reactive maintenance often leads to high repair costs. By contrast, excessive maintenance activities in preventive maintenance adversely induce unnecessary operational costs. With the emergence of technologies such as internet of things, big data, and artificial intelligence, predictive maintenance which involves data-driven methods is capturing huge attention in maintenance research and application. Within predictive maintenance, predicting the remaining useful life (RUL) of equipment using massive monitoring sensor data is considered one of the most important tasks. Deep learning has been shown to have superior performance and effectiveness in RUL prediction. In this thesis, several deep learning architectures are reviewed as well as their applications in RUL estimation. C-MAPSS dataset and a benchmark of several state-of-the-art papers are given for performance comparison. This benchmark suggests a future direction for RUL prediction research on hybrid deep learning models.

Keywords: Predictive maintenance, deep learning, remaining useful life

Preface

I want to thank Alexander Nikitin for his excellence guidance.

Otaniemi, 13.09.2020

Vinh Nguyen

Contents

Weekly timeline from week 24 to week 35

Week	Time	Task
24	08/06 - 14/06	12/06 Final research plan 14/06 First draft
25	15/06 - 21/06	Write autoEncoder
26	22/06 - 28/06	Write convolution neural network
27	29/06 - 05/07	Write recurrent neural network
28	06/07 - 12/07	Write hybrid network
29	13/07 - 19/07	Write transfer learning
30	20/07 - 26/07	Write generative neural network
31	27/07 - 02/08	Review previous writing
32	03/08 - 09/08	Add data description & benchmark Add conclusion 09/08 Second draft
33	10/08 - 16/08	Finalize the whole thesis
34	17/08 - 23/08	Prepare presentation
35	24/08 - 30/08	27/08 Final presentation 28/08 Final submission

1 Introduction

Maintenance is considered to be one of the most critical tasks in an industrial system. Each hour of downtime due to maintenance activities can result in a substantial loss [1, 2]. Furthermore, breaks in the highly critical component of a machine or system can lead to severe damage.

In order to avoid those risks, maintenance operation is inevitable. Companies and manufacturing enterprises have invested heavily in maintenance resources and procedures to ensure a constant operation and mitigate the risk of a system outage. Traditional maintenance programs deployed by different manufacturers vary in method and scale, but they typically fall into two main categories: reactive maintenance and preventive maintenance [3].

Reactive Maintenance (RM): In RM, also known as run-to-failure maintenance, equipment is allowed to operate until broken or to the end of its life. After that, maintenance is performed to restore the normal working condition of the equipment or replace it with a new one if necessary. This strategy has the advantage of utilizing all the useful life of the equipment. However, RM faces many challenges. In order to quickly react to all possible failures, a large spare part inventory or an immediate equipment delivery is required. Moreover, overtime labor and long system breakdown significantly contribute to the total cost.

Preventive Maintenance (PM): PM is conducted in a time-based fashion so that maintenance activities are scheduled with a predefined time interval. In this method, faults or breakdowns of machines and equipment are notably reduced due to regular examination or repair. Nevertheless, PM often leads to unnecessary actions and causes overhead in operational costs. [1]

Given the shortcomings of the above methods, there is an urgent demand for a more efficient maintenance method that can minimize the operational cost by conducting PM less frequently and maximize the reliability of the system by avoiding as much RM as possible. This can be achieved if potential faults and remaining useful life (RUL) of equipment or machines can be predicted in advance so that necessary interventions could be conducted effectively. This concept serves as the backbone of predictive maintenance (PdM), which has existed and has been studied for many years, but only with the emerging advanced technologies such as internet of things (IoT), clouds, and artificial intelligence (AI) that enable companies to utilize the full potential of PdM [4].

Massive amounts of data have been collected from industrial machinery and equipment by complex networks of different types of sensors to monitor the health condition of the system. Then, many machine learning (ML) techniques including Logistic Regression, Decision Tree, K-nearest Neighbor, Support Vector Machine, and Artificial Neural Network have been proposed and applied in PdM, and these methods delivered decent results. Nevertheless, traditional ML faces several limitations. Many ML techniques rely on handcrafted features from data that can be hard and time-consuming to derive. Moreover, these ML are linear in nature, making it impossible to capture non-linear and complex structures in industrial data.

In recent years, Deep Learning (DL) has been emerging as a powerful method

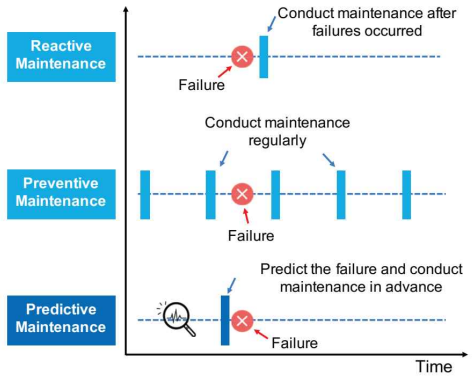

Figure 1: Maintenance plans of RM, PM, PdM [3].

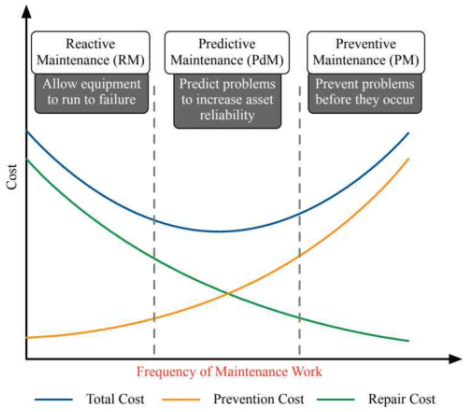

Figure 2: Comparison of RM, PM, and PdM on the cost and frequency of maintenance work [3].

that has been successfully applied to difficult tasks, such as image recognition, speech synthesis, and decision making. DL has the ability to automatically learn useful representation from data, relieve the burden of manual feature extraction. DL is also capable of capturing complex non-linear relationships, which enhances the classification and prediction power of the model.

With such appealing power of DL, researchers and engineers are rushing to apply DL into PdM. Within PdM, there are several key tasks including fault diagnosis and prognosis. Within the scope of this thesis, prognostics, and health management (PHM) is the primary focus. Specifically, prognostics is the process of accessing the future reliability of equipment or system, which is usually achieved via predicting their remaining useful life (RUL). In Section 2, a wide range of DL architectures and their applications in PHM are covered. Section 3 introduces a benchmark dataset as well as the performance of several reviewed methods.

2 Deep Learning Methods & Use Cases

2.1 AutoEncoder (AE)

2.1.1 Overview

Real-world data usually have high dimensions, which hinder machine learning algorithms from effectively learning from these data and performing well on downstream tasks. This is notoriously known as "Curse of dimensionality" [5]. Given such a problem, it is necessary to develop methods to reduce the number of dimensions of raw data but still preserve useful information in the reduced data. Autoencoder (AE) is a deep learning technique aiming at projecting the raw data into a latent space. AE is trained to ensure that the projection preserves most of the useful features and removes useless information from the original data. The obtained latent represen-

tations are exploited for further tasks, such as classification and regression. The general architecture of an autoencoder consists of two components: an encoder that maps the input data into a high-level representation, and a decoder that reconstructs the original data from the high-level features produced by the encoder. The mapping functions of encoder and decoder are usually modeled by two neural networks. In order to learn an efficient representation of the data, AE tries to minimize the dissimilarity between the reconstruction and the original input. However, there is a chance that the network will try to learn an identity function, which is trivial. To avoid a scenario, the size of the hidden layer is usually smaller than that of the input layer. This small size hidden layer can be seen as a bottleneck layer that forces the model to estimate a meaningful compression function. There are many variants of AE that address the problem of trivial identity mapping and also enable deeper model for better representation.

Sparse autoencoder (SAE): Instead of using bottleneck layers to constrain the network to learn to compress data, SAE imposes sparsity constraint on a hidden layer by forcing the average activation of a neuron over the training data to be close to zero. With this sparsity constraint, the remaining neurons having relatively large activation are shown to carries useful information. SAE achieves this by adding an additional objective to the loss function of AE:

$$\sum_{j=1}^{s_2} KL(\rho||\hat{\rho}) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j}$$

where ρ is a sparsity parameter that is close to zero, $\hat{\rho_j} = \frac{1}{m} \sum_{i=1}^{m} [a_j^{(2)}(x^{(i)})]$ is the average activation over the training set of j^{th} hidden neuron, and s_2 is the number of neurons in a hidden layer [6].

Denoising autoencoder (DAE): DAE takes a different strategy for learning informative features. DAE does not impose constraints on representation by bottle-neck or sparsity, but it tries to reconstruct the original data from the corrupted version of it. The authors of DAE argued that while learning to denoise, DAE is able to capture representation that is robust and stable under noisy and corrupted input, as well as implicitly realizing the underlying manifold of the input distribution. There are three common choices for data corruption functions: additive isotropic Gaussian noise where Gaussian noise is added to the data, masking noise where a random fraction of the data is set to zero, and salt-and-pepper noise where a random fraction of the data is forced to either their minimum or maximum possible value [7].

Contractive autoencoder (CAE): CAE learns to capture only important structures in the data by enforcing the gradient of the hidden activation with respect to the input signal to be close to zero. This technique makes the obtained hidden representations become insensitive to the variation in the input data, which contradicts the objective of AE that tries to capture all variation in the input. By optimizing the original objective and the proposed regularization term simultaneously, only crucial patterns are realized by the network. CAE imposes this regularization by adding the Frobenius

norm of the Jacobian of hidden activation with respect to the input [8].

$$\|J_f(x)\|_F^2 = \sum_{ij} \left(\frac{\partial h_j(x)}{\partial x_i} \right)^2$$

Stacked autoencoder (SAE): several AE can be stacked on top of each other to create a deeper network. Subsequent layers learn higher-level representations from the output features of previous layers. This enhances the ability to capture non-linear dependency in the data and yields better features. There are two main procedures to train SAE: the first one is similar to greedy layer-wise unsupervised pre-training, and the other is end-to-end training where the whole SAE is constructed and trained all layers at the same time.

2.1.2 RUL Prognosis

In industrial settings, a large swarm of sensors is deployed at different parts of the system to monitor its health condition and behavior. Some individual components even has multiple sensors attached to it. As a result, the collected data usually have high dimensions. AE is applied to address this problem, by transforming these high-dimensional data into a latent representation that has fewer dimensions.

Ren et al. [9] employed AE to compress time-domain vibration sensor data for RUL estimation. However, the method still relies on artificial features to make predictions, which might not exploit the full potential of deep learning. Ma et al. [10] proposes an architecture that first train a two-layer AE where the first layer is a denoising AE responsible for extracting robust features as well as removing noisy information intrinsic to sensor readings, and the second layer is a sparse autoencoder that induces sparsity on latent representation to learn the most sensitive degradation features. Both of the layers are pretrained using a greedy layer-wise unsupervised learning technique. The output of SAE is used to train a linear regression model for RUL prediction. Finally, SAE and LR are integrated and fine-tuned using supervised learning. Instead of using a single model for the entire degradation process, Xia et al. [11] classified the data into different health states and then trained a separate ANN model for each of the states to make RUL decision. In this proposed method, denoising AE was used as a feature extractor for health state classification.

2.2 Convolutional Neural Network (CNN)

2.2.1 Overview

Convolutional neural network (CNN) in the last few years has achieved dramatic success in many fields. From LeNet with superior classification accuracy on MNIST hand-written digits database that began the era of CNN [12], to the recent success in large-scale image classification ImageNet competition, where AlexNet [13], VGG [14], GoogLeNet [15], and ResNet [16] have proved the power of CNN. Going further, Redmon et al. [17] proposed You Only Look Once (YOLO) algorithm that enables the ability to detect several classes of objects in real-time, which immensely contributes

to the applications of object tracking or autonomous vehicles. Ronneberger et al. [18] proposed U-Net that is highly efficient in semantic segmentation, and the great result in segmenting biomedical cells. Not just be limited to image processing, 1-D CNN shows tremendous potential in modeling temporal dependency. WaveNet [19] uses dilated 1-D CNN as a generative model for human speech synthesis. The power of CNN can be addressed to 3 major characteristics:

1. Sparse Interaction: the size of the kernels are smaller than the size of the input. This substantially reduces the number of parameters of the network, which stabilize the statistical efficiency as well as decrease the number of operations.

2. Parameter Sharing: the same set of linear transformations of small local regions is applied to the whole input data. This also contributes to the reduction of memory requirements.

3. Equivariant Representation: the changes in the input are reflected similarly in the output.

[20].

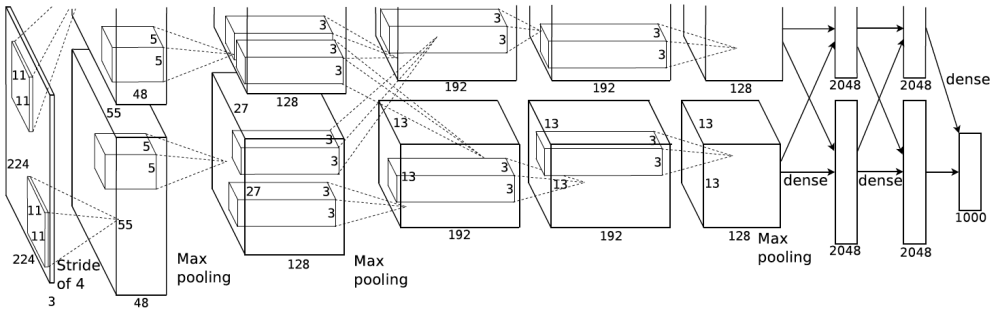

Figure 3: Example of CNN architecture. Here is the architecture of AlexNet [13].

Additionally, the recent advancement in parallel computing hardware such GPU and TPU has provided a huge speedup in training and deployment of CNN models. This facilitates the fast development of CNN.

2.2.2 RUL Prognosis

Khan et al. [21] transformed data into a 2-D format where one dimension was the number of features from different sensor readings and one dimension was the time sequence. The formatted data is fed to 1-D CNN to extract useful features, then an MLP was used on these learned features to predict RUL. The authors argued that the spatial relationship between different sensor reading series is not important, so although the data is 2-D, the convolutional operation is performed in 1-D fashion. In [22], Yang et al. used two CNN models to first identify the incipient fault point and then estimated RUL using the sensor data after the IF point. Li et al. [23] took the same approach with 1-D CNN, but instead of having only kernels of the same size at

each layer, three kernels of different sizes were used to extract features at different scales. The activated outputs of these kernels were added to form a single output. This method was represented as a multi-scale block, and combination of this block produces the proposed multi-scale deep CNN. To exploit the power of deep network, Wen et al. [24] utilized skip connection and residual learning from ResNet. Then an ensemble of k ResCNN models is composed to further improve the accuracy and robustness of the method. Moving on from 1-D CNN, Yoo et al. [25] used continuous wavelet transform to convert raw vibrational data to 2-D time-frequency feature images. After that, deep CNN was applied to estimate the health index and RUL.

2.3 Recurrent Neural Network (RNN)

2.3.1 Overview

Recurrent neural network (RNN) is a class of neural network architecture that is specialized in modeling sequential data. Unlike MLP where each neuron only works with the current input, RNN's neuron also takes into account its own output combined with the current input to produce the next output. With this mechanism, the obtained representation at each step contains information from all previous time steps, which enable the ability to capture temporal dependency in the data. In order to optimize RNN model, backpropagation through time (BPTT) was proposed [26] to account for the sequential nature of RNN. Unfortunately, RNN model usually does not work well in practice due to the exploding/vanishing gradient problem [27]. The longer the sequence that needs to be modeled, the more severe the issue is, making capturing long-term dependency difficult.

In order to address this problem, long short term memory (LSTM) and gated recurrent unit (GRU) were proposed where they pursue gated mechanism. LSTM was published first and then GRU was proposed as a streamlined version of LSTM with lower computational complexity and comparable performance. Both of these units learn to maintain critical dependencies and forget irrelevant features with various types of internal gates. With these gates working together, the general effect is the flexibility in modeling sequential data and ease of capturing long-term dependencies. Various variants of RNN such as multi-layer RNN, bidirectional RNN and attention-based RNN have been proposed to pushing the capability of these methods further. With their marvelous ability, LSTM and GRU have been applied to a wide range of applications where sequential data are essential like language modeling, machine translation, demand forecasting, speech modeling, anomalies detection, etc.

2.3.2 RUL Prognosis

Sensor readings in monitoring systems contain historical information about the degradation process as well as the current health stage of an equipment or machine. Realizing the remarkable ability to handle sequential data, many researchers have applied RNN family to exploit temporal dependency in these data to support PdM activities. Guo et al. [29] proposed a three-stage framework with the first stage was extracting 14 classical signal features, the second stage was selecting the most

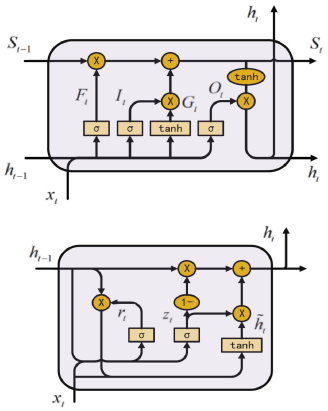

Figure 4: Architectures of LSTM and GRU [28].

informative features using correlation and monotonicity metrics, and the final stage was modeling RUL process via LSTM network. Wu et al. [30] utilized the vanilla form of LSTM on feature vectors consisting of original data plus 3 other handcrafted dynamic features to better address the problem of multiple operation modes and working conditions. Chen et al. [28] contributed with a 2-stage framework, which the first stage used kernel principal component analysis (KPCA) to reduce the number of dimensions and extract nonlinear features, then the second stage fed these features through GRU to capture the degradation process and estimate RUL. In [31], Zhao et al. proposed a framework that first used complete ensemble empirical mode decomposition to compose the best trend features reflecting the overall time sequence of degradation, the learned a RUL estimation model from these extracted features with LSTM network. Although these methods have shown significant results with low MSE, they still relied on many data preprocessing techniques to obtain handcrafted features, which required expertise and prior knowledge to achieve a good result. Instead of using manually constructive features, Wang et al. [32] used multi-layer LSTM on raw measurement data to predict RUL. A notable difference was that differential evolution (DE) algorithm was used instead of using gradient-based algorithms to optimize model parameters. Wang et al. [33] combined bidirectional LSTM model with attention mechanism to enhance RUL estimation efficiency, and DE algorithm was also applied for parameter optimization.

2.4 Hybrid Network (HN)

2.4.1 Overview

As reviewed earlier, each of the DL architectures has its own advantage. Most of the published papers only focus on one particular type of network. Nevertheless, there is an increasing trend in combining these architectures together so that they can complement each other and result in more robust models.

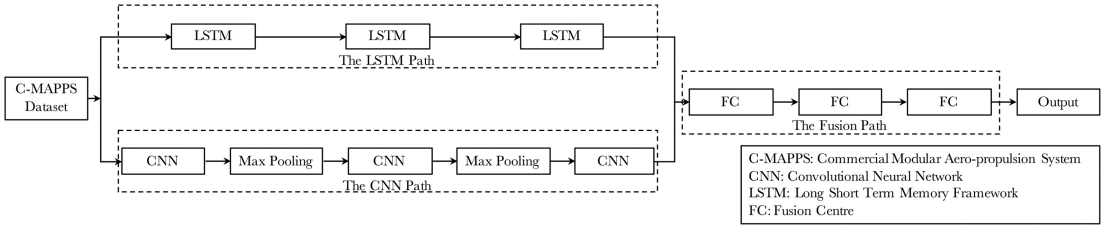

Figure 5: Example of hybrid network [34].

2.4.2 RUL Prognosis

Chen et al. [35] first used a multi-scale convolutional neural network (MSCNN) model as a pre-train step to extract local and global features from temporal data. The pre-trained MSCNN model was frozen to keep the parameters constant, then extracted features from this model are fed as input to CNN-LSTM model to utilize the historical conditions for early RUL estimation. Wang et al. [36] contributed a model that combines convolution, LSTM, and fully connected layers to better capture the temporal dependencies in the data in estimating RUL. Additionally, variational inference was applied to account for uncertainty and provide probabilistic RUL prediction. In [34], Dulaimi et al. proposed a RUL estimation model with two parallel paths: one path was CNN to capture spatial relations and one path was LSTM to exploit temporal dependencies. Features extracted from both of the two branches were fused by fully connected layers for final RUL estimation. Al-Dulaim et al. [37] took a similar approach with parallel path, with the addition of Gaussian noise layers throughout the architecture to discourage the model to memorize the training data and improve generalization.

2.5 Generative Adversarial Network (GAN)

2.5.1 Overview

Generative adversarial network, which was proposed by Ian Goodfellow [38], is undoubtedly one of the most impactful inventions in DL research. The concept of GAN is based on game theory, which involves two players in a mimimax game. In this game, a generator G aims at generating fake data as close to real data as possible, while a discriminator D tries to distinguish between real data and fake data. Generator G estimates a non-linear mapping from a latent noise distribution to the real data distribution, while discriminator D represents a binary classifier given both real and fake data. GAN learning process aims at reaching Nash Equilibrium, where synthetic data produced by the generator become indistinguishable to the discriminator. Specifically, the loss function can be derived as follow:

$$\min_{G} \max_{D} V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z))] \qquad (1)$$

where p_{data} is the data distribution, and p_z is the noise prior distribution.

GAN architecture brings tremendous advantages to both supervised and unsupervised learning. First, real data distribution is implicitly inferred during training

enabling the generation of realistic data that could be utilized for imbalanced/missing data issues. Second, GAN learns latent representation that could be utilized in downstream tasks. Furthermore, GAN could potentially be applied to domain adaptation. Although showing marvelous qualifications, GAN still has its own limitations. GAN notoriously suffers from mode collapse, in which the generator only learns to produce data of a subset of modes of the real data distribution to fool the discriminator, which limits the diversity of synthetic data [39]. Next, the training of GAN suffers from instability where it is difficult for the two models to reach Nash Equilibrium [40, 41]. Another challenge in GAN is that there is no reliable quantitative metrics for evaluating the quality of data generated by GAN [42].

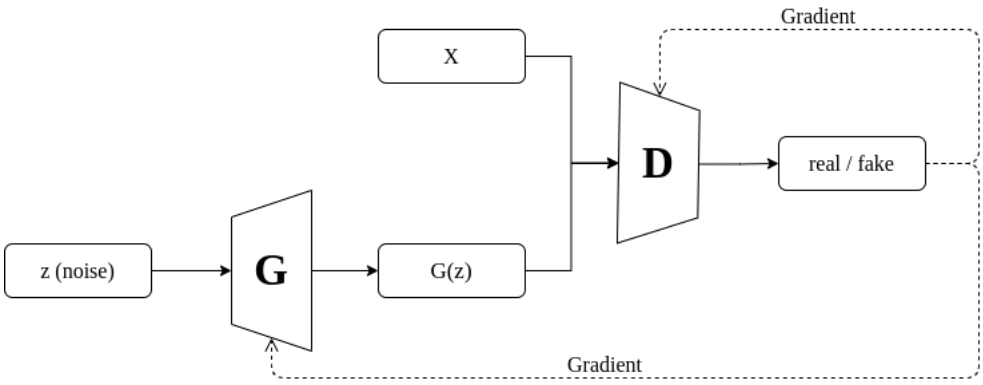

Figure 6: Vanilla GAN architecture. Generator takes in random noise to generate fake data. Discriminator tries to distinguish real data from fake data.

2.5.2 RUL Prognosis

With all the innovation that GAN brings to the AI/DL research, it is inevitable that this method is applied in the field of PHM. In [43] Khan et al. applied GAN on bearing vibration acceleration signals to model the degradation process distribution, which enables sampling signal trajectories having similar behavior to the original data. The proposed framework produced a decent result on univariate signals, showing the potential of extension to multivariate signals. Li et al. [44] utilized GAN architecture in two different stages. First, the author used GAN to estimate the distribution of healthy data, and the feature extracting layers of the discriminator was utilized to identify the first predicting time (FPT) where the degradation process begins. Next, adversarial learning was utilized to induce capturing domain invariant features, which effectively yielded adapted feature space for both training and testing data.

2.6 Transfer Learning (TL)

2.6.1 Overview

A wide range of DL methods have been proposed in PdM and showed superior performance on different tasks. However, these methods are designed with the

assumption that the training data and test data come from the same distribution. DL methods only learn relationships and dependencies in this distribution and perform really well if they are tested with data of the same distribution. This becomes a major challenge for DL to be deployed to a diverse application domain where machines and tools often operate on a wide range of modes and working conditions. In order to ensure the generalization of the DL models, a massive amount of data need to be acquired. This requirement often is impractical and even impossible in real-world scenarios because the data acquisition process is usually expensive, time-consuming, and labor demanding. Furthermore, many pieces of machines and equipment are not allowed to run until failure, or their degradation processes might happen in a long period of time, making it difficult to obtain their degrading progression data [45]. With these limitations, there exists a demand to develop solutions to address the problem of a lack of data. Many efforts have been proposed, and these works try to transfer knowledge from the source domain to the target domain. According to Tan et al. [46], deep transfer learning techniques are classified into four categories:

1. Instances-based: this method is developed based on the assumption that despite coming from different distributions, a portion of source instances that can be useful in learning target domain tasks. This utilization of source domain data is enabled by appropriate weights. Dai et al [47] proposed TrAdaBoost which aims at removing distant source instances and construct a set of source instances that are similar to target instances by re-weighting.

2. Mapping-based: the purpose of this technique is to find a latent space can effectively represent both the source and target domain, which facilitates subsequent learning tasks. One of the widely used approaches is minimizing the distance of the latent representation distributions of the two domains. Maximum mean discrepancy (MMD), which is a function to measure the distance between two distribution, was proposed as an objective to optimize to induce the creation of common representation space [48].

3. Network-based: more and more large-scale datasets have been published and serve as valuable sources of data. Models trained on these datasets are able to exploit several common features that are transferable among different tasks. ImageNet [49] is a massive image database with more than 14 million hand-annotated images covering a wide range of categories. Models trained on ImageNet such as AlexNet, ResNet, MobileNet perform really well at extracting useful features for visual problems. A subset of layers in these pretrained models can be utilized to other tasks that involve visual understanding such as image/video captioning and semantic segmentation. Recent published BERT pretrained on the English Wikipedia database displays superior capability in capturing complex relationships in natural language, which can be utilized as the backbone for related tasks such as sentiment classification, language translation, and question answering. Two common strategies for using network-based transfer learning are using pretrained models as a fixed feature extractor

and fine-tuning the whole model with target domain data.

4. Adversarial-based: this method is influenced by GAN, where models are encouraged to produce indistinguishable features between source and target domain distributions, leading to improvement on generalization. Typically, adversarial adaptation involves a feature extractor and a domain discriminator simultaneously. Under the guidance of the discriminator which trying to distinguish the origin of the input data, the feature extractor is forced to produce domain-invariant features between source and target distributions [44] [50].

2.6.2 RUL Prognosis

Zhang et al. in [45] proposed TL framework where a BiLSTM model was pretrained on large source dataset that was different but related. After that, the model was further fine-tuned with the limited target dataset. In [51], Sun et al. used SAE combined with TL techniques to address the data scarcity problem. The authors trained an SAE to map the source data to a feature space and transferred learned parameters to a second SAE. The new SAE was trained on target dataset with feature transfer regularization that used KL divergence to minimize the distance between the source and target feature spaces. Costa et al. [52] applied adversarial learning in learning domain-invariant representations for prognosis. The framework aimed at both minimizing source regression loss and maximizing domain classification loss on the same network, which produced domain invariant representations that are useful for regression tasks. Li et al. [53] applied adaptive batch normalization for transfer learning. First, CNN model was trained on the source domain to learn a feature extractor, then parameters in all convolutional layers were frozen, and only parameters of batch normalization layers were tuned further with target data, allowing source latent space adapt to target domain.

3 Experiments & Benchmark

In this section, a comparison is made to provide insight into the performance of several PHM methods. First, a popular dataset in RUL prognosis is reviewed to serve as a base for benchmarking. Two performance metrics are introduced to quantify models' performance.

3.1 NASA C-MAPSS dataset

The turbofan engine degradation dataset, which was developed by NASA, was simulated from commercial modular aero-propulsion system simulation (C-MAPSS). The dataset consists of four subdatasets: FD001, FD002, FD003, FD004, where each subset contains several multivariate time series representing sensor reading of different turbo engines with different initial wear and working conditions. Detail description for these datasets is available in Table 1. Each subset is divided further

into a training set and a test set. Training set consists of full lifespan of engines whose early states are healthy and degrade gradually until complete failure. In the test set, sensor trajectories of engines end at some point prior to failure, and the RUL of these engines are given for prognosis tasks. Each sensor observation series contains 26 variables: the first 2 variables indicate engine number and current running cycle, the next 3 are operational variables, and the remaining 21 variables are sensor measurements. Note that some of the sensors show no trend in their observation trajectories, meaning that they don't convey meaningful information to facilitate RUL estimation. Therefore, as in several papers [23, 24, 34, 37] a subset of useful sensors are selected, and their indices are 2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20, and 21.

Dataset	Engine fleet			
	FD001	FD002	FD003	FD004
Train trajectories	100	260	100	249
Test trajectories	100	259	100	248
Operation modes	1	6	1	6
Fault modes	1	1	2	2

Table 1: The C-MAPSS Dataset Details

3.1.1 Data pre-processing

In C-MAPSS dataset, measurements are in different scales, which is shown to be difficult for machine learning algorithms. Therefore, transforming data to the same scale will be beneficial for learning process and improve performance. In the field of PHM, data standardization and normalization are the the most common techniques.

$$Standardization : x_i^j = \frac{x_i^j - \mu^j}{\sigma^j} \tag{2}$$

$$Normalization : x_i^j = \frac{2(x_i^j - x_{min}^j)}{x_{max}^j - x_{min}^j} - 1 \tag{3}$$

where x_i^j, x_{min}^j, x_{max}^j, μ^j, σ^j denotes i^{th} time point, minimum, maximum, mean, and standard deviation of j^{th} sensor, respectively. These statistics are calculated from training datasets, and they are applied to scale both the training datasets and test datasets.

3.1.2 RUL target function

Predicting RUL bears many resemblances to the regression problem. However, in industrial settings, the desired numerical RUL target for each data point is challenging to obtain without an accurate physical-based model reflecting the properties and behaviors of a machine or system. As a heuristic, piece-wise linear degradation

function was proposed in [54] and adopted in several PHM papers. The motivation behind this target function is that the engine unit is healthy in its initial operation, and only starts degrading after several working cycles. It is reasonable to assign constant RUL in the early phase, then a linearly degrading function is applied after that.

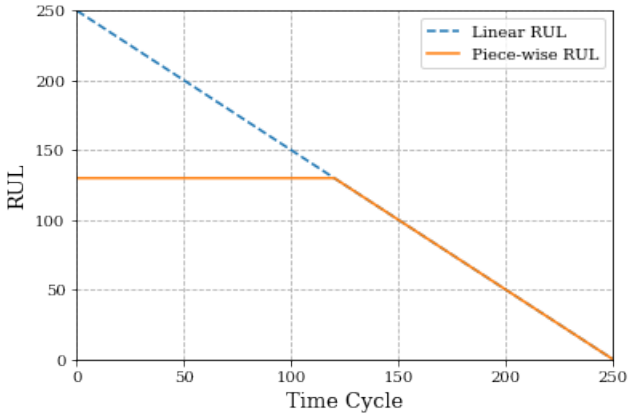

Figure 7: The illustration of piece-wise linear RUL target function.

3.1.3 Evaluation metrics

In most of PHM papers, two performance metrics: scoring function and root mean square error (RMSE) were used to evaluate their proposed methods.

1. PHM08 score: the scoring function has been employed by the International Conference on Prognostics and Health Management Data Challenge (PHM08). The function is given by:

$$S = \sum_{i=1}^{N} s_i \text{ where } s_i = \begin{cases} e^{-\frac{h_i}{13}} - 1 & \text{for } h_i < 0 \\ e^{\frac{h_i}{10}} - 1 & \text{for } h_i \geq 0 \end{cases} \tag{4}$$

   where S is the final score, N is the total number of sample, and $h_i = RUL_{pred} - RUL_{real}$ This evaluation metric give more penalty toward late prediction rather than early prediction. The reason is that in the industrial environment, late prediction is more harmful where the production lines or critical system processes can be halted, which potentially leads to severe consequences. Meanwhile, early prediction only induces inefficient maintenance costs.

2. RMSE: this is a popular performance metric in regression-like tasks. RMSE gives equal weight to both early and late prediction. The formula is illustrates as follow:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} h_i^2} \tag{5}$$

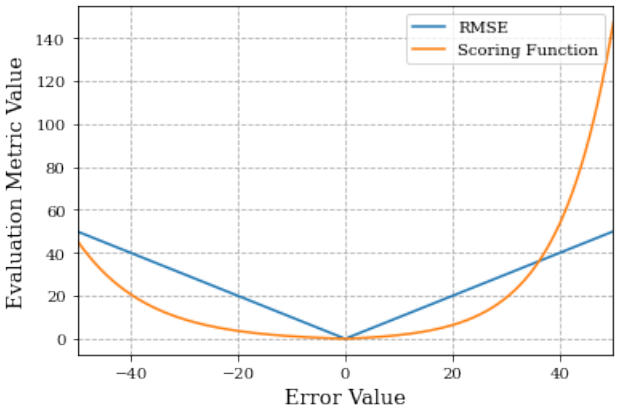

Figure 8: The comparison between scoring function and RMSE function.

3.2 Benchmarks

Table 2 summarizes the performance of several proposed methods. As clearly shown, hybrid architectures that combine advantages of various DL classes consistently outperform standalone frameworks. This comparison suggests a promising direction for future PHM researches as well as practical applications to emphasize more on hybrid networks.

Method	FD001		FD002		FD003		FD004	
	PHM08	RMSE	PHM08	RMSE	PHM08	RMSE	PHM08	RMSE
D-LSTM [55]	338.00	16.14	4450.00	24.49	852.00	16.18	5550.00	28.17
HD-LSTM [56]	-	14.57	-	23.20	-	14.92	-	28.72
HDNN [34]	245.00	13.017	1282.42	15.24	287.72	12.22	1527.42	18.156
EnResCNN [24]	212.48	12.16	2087.77	20.85	180.76	12.01	3400.44	24.97
MS-DCNN [23]	196. 22	11.44	3747.00	19.35	241.89	11.67	4844.00	22.22
DSCN [57]	260.67	10.95	4367.56	20.47	246.55	10.62	5168.45	22.64
CNN+BiLSTM [58]	-	10.74	-	15.20	-	13.85	-	18.60
NPBGRU [37]	**191.80**	**10.44**	**899.76**	**14.65**	**197.46**	**10.59**	**1306.50**	**16.78**

Table 2: Performance comparison of 8 state-of-the-art methods on C-MAPSS dataset. Bold indicates best performance. Lower is better.

4 Conclusions

This thesis presents a comprehensive literature survey on the application of DL on PHM. The first section introduces three types of maintenance methods with the emphasis on PdM. The next section reviews various DL architectures and their corresponding research papers. The final section provides an overview of the C-MAPSS dataset as well as a benchmark table for several state-of-the-art methods in RUL estimation.

References

[1] R. Keith. Mobley. *An introduction to predictive maintenance*. Butterworth-Heinemann, 2002.

[2] Wu, Tsui Shaomin, Chen Kwok L., Zhou Nan, Hai Qiang, Wang Yizhen, and Wenbin. Prognostics and Health Management: A Review on Data Driven Approaches. *Mathematical Problems in Engineering*, 2015, 2015.

[3] Yongyi Ran, Xin Zhou, Pengfeng Lin, Yonggang Wen, and Ruilong Deng. A Survey of Predictive Maintenance: Systems, Purposes and Approaches. *arXiv e-prints*, page arXiv:1912.07383, December 2019.

[4] Yongyi, Zhou, Xin, Lin, Pengfeng, Wen, Yonggang, Deng, and Ruilong. A survey of predictive maintenance: Systems, purposes and approaches. *arXiv.org*, Dec 2019.

[5] I K Fodor. A survey of dimension reduction techniques. Technical report, Lawrence Livermore National Lab., CA (US), 2002.

[6] Andrew Ng. Sparse autoencoder. URL: https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf. (2020/08/08).

[7] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010.

[8] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. pages 833–840, 2011.

[9] Lei Ren, Yaqiang Sun, Jin Cui, and Lin Zhang. Bearing remaining useful life prediction based on deep autoencoder and deep neural networks. *Journal of Manufacturing Systems*, 48:71–77, 2018.

[10] Jian Ma, Hua Su, Wan-Lin Zhao, and Bin Liu. Predicting the remaining useful life of an aircraft engine using a stacked sparse autoencoder with multilayer self-learning. *Complexity*, 2018:1–13, 2018.

[11] Min Xia, Teng Li, Tongxin Shu, Jiafu Wan, Clarence W. De Silva, and Zhongren Wang. A two-stage approach for the remaining useful life prediction of bearings using deep neural networks. *IEEE Transactions on Industrial Informatics*, 15(6):3703–3711, 2019.

[12] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. pages 2278–2324, 1998.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. 2014. cite arxiv:1409.4842.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. 2015. cite arxiv:1512.03385Comment: Tech report.

[17] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, page 234–241, 2015.

[19] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. 2016. cite arxiv:1609.03499.

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[21] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 2020.

[22] Boyuan Yang, Ruonan Liu, and Enrico Zio. Remaining useful life prediction based on a double-convolutional neural network architecture. *IEEE Transactions on Industrial Electronics*, 66(12):9521–9530, 2019.

[23] Han Li, Wei Zhao, Yuxi Zhang, and Enrico Zio. Remaining useful life prediction using multi-scale deep convolutional neural network. *Applied Soft Computing*, 89:106113, 2020.

[24] Long Wen, Yan Dong, and Liang Gao. A new ensemble residual convolutional neural network for remaining useful life estimation. *Mathematical Biosciences and Engineering*, 16(2):862–880, 2019.

[25] Youngji Yoo and Jun-Geol Baek. A novel image feature for the remaining useful lifetime prediction of bearings based on continuous wavelet transform and convolutional neural network. *Applied Sciences*, 8(7):1102, Aug 2018.

[26] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[27] Roger Grosse. Exploding and Vanishing Gradients. URL: http://faculty.cse.tamu.edu/ajiang/636_VEG.pdf. (2020/08/08).

[28] Jinglong Chen, Hongjie Jing, Yuanhong Chang, and Qian Liu. Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process. *Reliability Engineering System Safety*, 185:372–382, 2019.

[29] Liang Guo, Naipeng Li, Feng Jia, Yaguo Lei, and Jing Lin. A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing*, 240:98–109, 2017.

[30] Yuting Wu, Mei Yuan, Shaopeng Dong, Li Lin, and Yingqi Liu. Remaining useful life estimation of engineered systems using vanilla lstm neural networks. *Neurocomputing*, 275:167–179, 2018.

[31] Sen Zhao, Yong Zhang, Shang Wang, Beitong Zhou, and Cheng Cheng. A recurrent neural network approach for remaining useful life prediction utilizing a novel trend features construction method. *Measurement*, 146:279–288, 2019.

[32] Fu-Kwun Wang, Xiao-Bin Cheng, and Kai-Chun Hsiao. Stacked long short-term memory model for proton exchange membrane fuel cell systems degradation. *Journal of Power Sources*, 448:227591, 2020.

[33] Fu-Kwun Wang, Tadele Mamo, and Xiao-Bin Cheng. Bi-directional long short-term memory recurrent neural network with attention for stack voltage degradation from proton exchange membrane fuel cells. *Journal of Power Sources*, 461:228170, 2020.

[34] Ali Al-Dulaimi, Soheil Zabihi, Amir Asif, and Arash Mohammadi. A multimodal and hybrid deep neural network model for remaining useful life estimation. *Computers in Industry*, 108:186–196, 2019.

[35] Zesheng Chen, Xiaotong Tu, Yue Hu, and Fucai Li. Real-time bearing remaining useful life estimation based on the frozen convolutional and activated memory neural network. *IEEE Access*, 7:96583–96593, 2019.

[36] Biao Wang, Yaguo Lei, Tao Yan, Naipeng Li, and Liang Guo. Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery. *Neurocomputing*, 379:117–129, 2020.

[37] Ali Al-Dulaimi, Amir Asif, and Arash Mohammadi. Noisy parallel hybrid model of NBGRU and NCNN architectures for remaining useful life estimation. *Quality Engineering*, 32(3):371–387, 2020.

[38] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. pages 2672–2680, 2014.

[39] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U. Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. pages 3308–3318, 2017.

[40] Martín Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. 2017.

[41] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *CoRR*, abs/1701.07875, 2017.

[42] Ali Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

[43] Sheraz Ali Khan, Alexander E. Prosvirin, and Jong-Myon Kim. Towards bearing health prognosis using generative adversarial networks: Modeling bearing degradation. *2018 International Conference on Advancements in Computational Sciences (ICACS)*, 2018.

[44] Xiang Li, Wei Zhang, Hui Ma, Zhong Luo, and Xu Li. Data alignments in machinery remaining useful life prediction using deep adversarial neural networks. *Knowledge-Based Systems*, 197:105843, 2020.

[45] Ansi Zhang, Honglei Wang, Shaobo Li, Yuxin Cui, Zhonghao Liu, Guanci Yang, and Jianjun Hu. Transfer learning with deep recurrent neural networks for remaining useful life estimation. *Applied Sciences*, 8(12):2416, 2018.

[46] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A Survey on Deep Transfer Learning. *arXiv e-prints*, page arXiv:1808.01974, August 2018.

[47] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. *Proceedings of the 24th international conference on Machine learning - ICML 07*, 2007.

[48] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Scholkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14), 2006.

[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[50] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[51] Chuang Sun, Meng Ma, Zhibin Zhao, Shaohua Tian, Ruqiang Yan, and Xuefeng Chen. Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing. *IEEE Transactions on Industrial Informatics*, 15(4):2416–2425, 2019.

[52] Paulo Roberto De Oliveira Da Costa, Alp Akçay, Yingqian Zhang, and Uzay Kaymak. Remaining useful lifetime prediction via deep domain adaptation. *Reliability Engineering System Safety*, 195:106682, 2020.

[53] Jialin Li, Xueyi Li, and David He. Domain adaptation remaining useful life prediction method based on adabn-dcnn. *2019 Prognostics and System Health Management Conference (PHM-Qingdao)*, 2019.

[54] Felix O. Heimes. Recurrent neural networks for remaining useful life estimation. *2008 International Conference on Prognostics and Health Management*, 2008.

[55] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2017.

[56] Min Xia, Xi Zheng, Muhammad Imran, and Muhammad Shoaib. Data-driven prognosis method using hybrid deep recurrent neural network. *Applied Soft Computing*, 93:106351, 2020.

[57] Biao Wang, Yaguo Lei, Naipeng Li, and Tao Yan. Deep separable convolutional network for remaining useful life prediction of machinery. *Mechanical Systems and Signal Processing*, 134:106330, 2019.

[58] Ikram Remadna, Sadek Labib Terrissa, Ryad Zemouri, Soheyb Ayad, and Noureddine Zerhouni. Leveraging the power of the combination of cnn and bi-directional lstm networks for aircraft engine rul estimation. *2020 Prognostics and Health Management Conference (PHM-Besançon)*, 2020.