2022 IEEE International Conference on Prognostics and Health Management (ICPHM) | 978-1-6654-6615-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICPHM53196.2022.9815692

A Framework for Generating Large Data Sets for Fatigue Damage Prognostic Problems

Anass Akrim[†‡], Christian Gogu[‡], Thomas Guillebot de Nerville[†], Paul Strähle[†],
Brondon Waffa Pagou[‡], Michel Salaün[†], and Rob Vingerhoeds[†]
[†]ISAE-SUPAERO, Université de Toulouse,
10 Avenue Edouard Belin, 31400 Toulouse, France
Email: {anass.akrim, michel.salaun, rob.vingerhoeds}@isae-supaero.fr
[‡]Institut Clément Ader (UMR CNRS 5312) INSA/UPS/ISAE/Mines Albi,
Université de Toulouse, 3 rue Caroline Aigle, 31400 Toulouse, France
Email: {anass.akrim, christian.gogu}@gmail.com

***Abstract*—Prognostics and Health Management (PHM) relies on the availability of large amounts of data for a given system and allows to analyse this data and to draw conclusions as to the health state of the system, the identification of faults and failures, as well as the calculation of the remaining useful life time. Often, it is expected that this data is labelled, *i.e.* that the data has been pre-analysed and that for each data point an exact information is available as to what it is about, when it was measured, etc. In reality, this is not always easy and this labeled data is not always available. For example, on aerospace structures, complete labeled data until the end of their lifetime are not usually available. This may hamper for example the use of Deep Learning (DL) techniques for Predictive Maintenance, as they rely on the availability of large amounts of labeled sensor data. In this paper a framework and associated code[1] is proposed to generate high dimensional data sets for a realistic fatigue damage prognostics problem, representative of fatigue cracks propagation in aeronautical fuselage panels. With this data, DL techniques can be trained, and we will illustrate this with a case study involving several of the most commonly used DL models to address failure prognostics.**

***Index Terms*—Prognostics and Health Management (PHM), Remaining Useful Life (RUL), Fatigue Damage Prognostics Problem (FDPP), Synthetic Dataset, Deep Learning.**

I. Introduction

Prognostics and Health Management (PHM) is a field of research and application, making use of past and present available information of an equipment in order to detect its degradation, diagnose its faults, predict and manage its failures [1]. Fatigue damage is defined as one of the major life-limiting factors for many structural components subjected to variable loadings in service (e.g. aircrafts during flight) [2]. Fatigue is the cause of various failure modes in aerospace, develops gradually and progressively grows to a critical size a_{crit}, leading to structural or system failure. The operating time before failure is commonly referred to as Remaining Useful Life (RUL) [3]. Fatigue monitoring and potential early identification of critical cracks approaching the critical size a_{crit} is a major challenge, with great potential in terms of improving the operational efficiency through the development of predictive maintenance strategies. Hence, prediction of fatigue life in structures is necessary, becoming one of the main issues in the field of operational safety.

Among Prognostics approaches, Data-Driven models have gained more and more attention in the PHM community, especially the latest Machine Learning (ML) techniques (notably Deep Learning (DL) techniques) [4]–[6]. DL has become a major and rapidly growing research direction, redefining state-of-the-art performances in a wide range of areas in recent years [7]. As more data becomes available in the engineering domain, there is a recent surge of interest in using Deep Learning in Prognostics and Health Management [8]. However, their effectiveness depends on the quantity and quality of available labeled data, which is generally difficult to acquire and often can be a time-consuming and expensive investment. Indeed in PHM, faults are rare and structures can be replaced before reaching failure; in Prognostics tasks, a label can constitute the RUL at each time step of measurements. Hence, data scarcity is becoming one of the most important challenges in PHM [9], [10], rendering it difficult to evaluate and compare the latest DL models in the research field.

Several PHM researchers have already attempted to address this challenge, proposing realistic synthetic data sets that have been collected and made publicly available by NASA's Prognostic Center of Excellence, such that for turbofan engines [3], bearings [11], batteries [12], etc. The reader may refer to [13] for more details on commonly used synthetic data sets in PHM. Although these open source data sets have been widely successful among the PHM community, to the best of the authors' knowledge, little to no research has directly aimed at proposing and making available an open source framework for generating large data sets specific to fatigue damage prognostic problems. Virkler et al. [14] proposed a fatigue crack growth dataset for fatigue damage prognostic problems, allowing several PHM researchers to evaluate data-driven approaches on it [15], [16]. However, the proposed datasets consist only of time series of structural crack length data and, according to [13], indirect sensory

[1]See https://github.com/ansak95/FrameworkFDPP

measurements (e.g. vibration, acoustic emissions, strains, etc.) are not provided, making it difficult to transfer to real life case applications where there is no direct access to crack length data but increasing amounts of indirect sensor data may instead be available.

In order to stimulate research on the applicability of the latest deep learning models to fatigue damage prognostics, notably in the aerospace domain, while awaiting real in service data, our paper proposes a framework and software code for synthetically generating large training data sets for a realistic fatigue damage prognostics problem (FDPP), simulating crack propagation in a fuselage panel, where the indirect sensory measurements correspond to mechanical strains. The crack growth is simulated based on Paris-Erdogan's crack growth model [17] and we consider strain data at various position (i.e. synthetic strain gauges) as output that will be used as sensor data. Due to the synthetic nature of this data set, it is possible to significantly vary the size of the training data sets, which may be particularly relevant for some deep learning approaches. The authors believe that the proposed framework and software code can help facilitate the development of deep learning algorithms for prognostic applications, and make them more easily transferable to real world applications. As illustration, some of the most common deep learning models have been applied to the generated data sets, including Recurrent Neural Network (RNN), Long short-term memory (LSTM), Gated Recurrent Unit (GRU), 1D-Convolutional Neural Network (CNN), and Temporal Convolutional Network (TCN), and we notably investigated the variation of the methods' performance with increasing labeled training data. All codes, both for the generation of the synthetic datasets and for the training of all the models are publicly available on https://github.com/ansak95/FrameworkFDPP.

The remainder of the paper is organized as follows. Section II describes the chosen approach to generate a synthetic data set simulating the crack growth in the considered structures. The degradation model used to generate the data set is presented in this section as well. Section III illustrates a realistic case study in order to investigate the applicability of DL methods in a prognostics problem based on a generated synthetic data set. Section III-A describes the data set generated. Section III-B presents the deep learning-based models investigated to illustrate this study and details the proposed RUL estimation strategy. The proposed metric for performance evaluation, used to rank the models investigated, are presented in this section as well. Section III-C presents and analyzes the results of the investigated deep learning-based approaches in this study. Finally, Section IV concludes the paper and identifies some research perspectives.

II. Framework

The proposed framework seeks to generate synthetic data sets of mechanical strain data (i.e. virtual strain gauges), simulating the crack growths in structures based on the Paris-Erdogan model. These synthetic data sets will be composed of labeled data, i.e. measurement sequences of structures until failure, where the label is the RUL of the structure at each time step of the strain measurements.

A. Simulation of the crack growth : a theoretical approach

1) Crack growth model: Fatigue crack propagation is modeled by the Paris-Erdogan's law [18]

$$\frac{da}{dk} = C(\Delta\sigma\sqrt{\pi a})^m \tag{1}$$

where a is the half crack size, k is the number of loading cycles, C and m are the empirical parameters of the Paris-Erdogan's law, and $\Delta\sigma$ is the difference between the minimum σ_{min} and maximum σ_{max} far field stress.

This law will be used to simulate the crack growth in the current setting to give the crack length at a given number of cycles, which will in turn be used to generate the deformation data on which the data-driven models will be trained. To get the crack length after k cycles, the analytical solution to Paris-Erdogan's law from [19] provided in Eq. 2 will be used.

$$a_k = \left[kC\left(1-\frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}} \tag{2}$$

where a_0 is the initial crack length.

The critical crack size a_{crit} that causes structural failure can be calculated by the empirical formula in Eq. (3).

$$a_{crit} = (\frac{K_I}{\Delta\sigma\sqrt{\pi}})^2 \tag{3}$$

where K_I is a conservative estimate of the fracture toughness in Mode I crack loading [20].

2) Strain fields around the fatigue crack: In order to obtain the strain field around the crack, we work under the assumption of a finite crack in an infinite plate under Mode I crack loading [20], assuming $\Delta\sigma = \sigma_{max} - \sigma_{min}$ with $\sigma_{min} = 0\ MPa$. In practice, the infinite plate assumption implies that the crack size must be much smaller than the size of the plate, which is typically verified for structures such as fuselage panels. A figure of a crack of length $2a$ and a loading of σ_{max} is given in Figure 1.

To calculate the displacements around the crack, a complex variable formulation along with a Westergaard approach is used [18], [20], [21]. In a Cartesian coordinate system (x, y), a complex variable z is introduced :

$$z = x + iy \tag{4}$$

The Airy stress function ϕ used in this approach is defined such that :

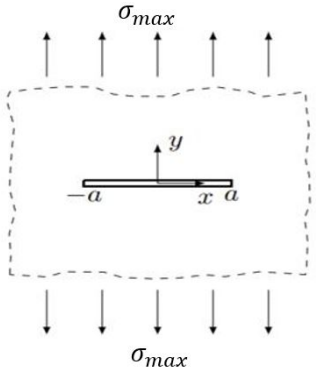

Fig. 1. Crack of length $2a$ in a plate under Mode I loading

$$\phi = \sigma_{max}\sqrt{z^2-a^2} - \sigma_{max}z + \text{const.} \quad (5)$$

$$\phi' = \frac{\sigma_{max}z}{\sqrt{z^2-a^2}} - \sigma_{max} \quad (6)$$

$$\phi'' = \frac{\sigma_{max}}{\sqrt{z^2-a^2}} - \frac{\sigma_{max}z^2}{(z^2-a^2)^{\frac{3}{2}}} \quad (7)$$

The stresses in a plane stress state, which is assumed here, can then be expressed as

$$\sigma_{11} = \Re(\phi') - y\Im(\phi'') \quad (8)$$

$$\sigma_{22} = \Re(\phi') + y\Im(\phi'') + \sigma_{max} \quad (9)$$

$$\sigma_{12} = -y\Re(\phi'') \quad (10)$$

where $\Re(.)$ and $\Im(.)$ are respectively the real part and the imaginary part of a complex number.

The corresponding strain state for this stress state are given by Hooke's law [22], which can be written under the plane stress assumption as :

$$\varepsilon_{11} = \frac{1}{E}(\sigma_{11} - \nu\sigma_{22}) \quad (11)$$

$$\varepsilon_{22} = \frac{1}{E}(-\nu\sigma_{11} + \sigma_{22}) \quad (12)$$

$$\varepsilon_{12} = \frac{1+\nu}{E}\sigma_{12} \quad (13)$$

$$\varepsilon_{33} = \frac{-\nu}{E}(\sigma_{11} + \sigma_{22}) \quad (14)$$

where E and ν are the elastic parameters, respectively Young's modulus and Poisson's ratio. A typical normal strains state is shown in Fig. 2.

Finally, the strain measurements ε of a strain gauge placed at the position (x, y) with an angle θ between the x-axis and the strain gauge measurements direction are given by Eq. 15.

$$\varepsilon = \varepsilon_{11}\cos(\theta)^2 + \varepsilon_{22}\sin(\theta)^2 + 2\varepsilon_{12}\cos(\theta)\sin(\theta) \quad (15)$$

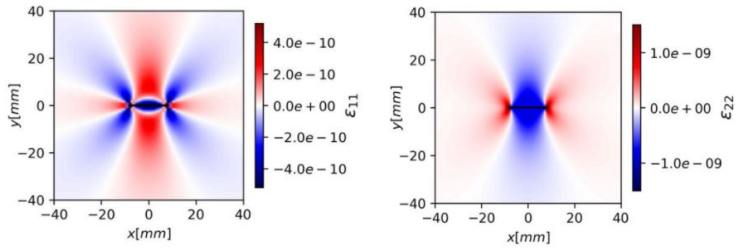

Fig. 2. Example normal strains state for a crack of length $2a$

B. Generation Algorithm

All data sets composed of strain measurements are generated using the previously defined crack growth model and, for initialization, each structure n is defined by three parameters that vary from structure to structure within the data set : the initial crack length $a_{0,n}$, and the two material parameters C_n and m_n generating the crack growth. Note that in this paper, we consider that C is distributed following a log-normal law, and $\log C$ and m are assumed to follow a multivariate distribution with a linear correlation coefficient ρ, based on the literature [23], [24]. The steps of the numerical implementation of the data set creation are summarized below, illustrated in Fig. 3:

1) For the n^{th} structure $(n = 1, ..., N)$, draw the samples of initial crack size and the Paris-Erdogan's law parameters respectively : $a_{0,n} \sim \mathcal{N}(\mu_{a_0}, \sigma_{a_0})$ and $(m_n, \log C_n) \sim \mathcal{N}(\mu_m, \sigma_m, \mu_{\log C}, \sigma_{\log C}, \rho)$.
2) Using the crack growth model described in Section II-A1, propagate the crack size of the n^{th} structure until it reaches the critical crack size a_{crit}. Every Δk cycles until failure, compute and collect the strain measurements at the n_g strain gauge positions according to Eq. 15.

In order to train prognostics models based on this strains dataset we also need the RUL at each cycle. For the n^{th} structure at cycle k, the remaining useful life $RUL_{n,k}$ can be computed such that $RUL_{n,k} = k_{crit} - k$ (where k_{crit} is the number of cycles for the crack to reach the critical size, such that $a_{k_{crit}} = a_{crit}$). An illustration of three generated sequences (i.e. three strain gauges) for a single structure until failure is given in Fig. 4.

III. Illustration

In this section, an illustration of the framework, the generated data sets and the use of Deep Learning (DL) techniques for the estimation of the Remaining Useful Lifetime (RUL) is provided.

A. Data Set

Starting from the analytical setup described in the previous section, a synthetic data set is generated containing the variations of the strains at $n_g = 3$ positions in the plate as a function of the number of cycles. This setup can

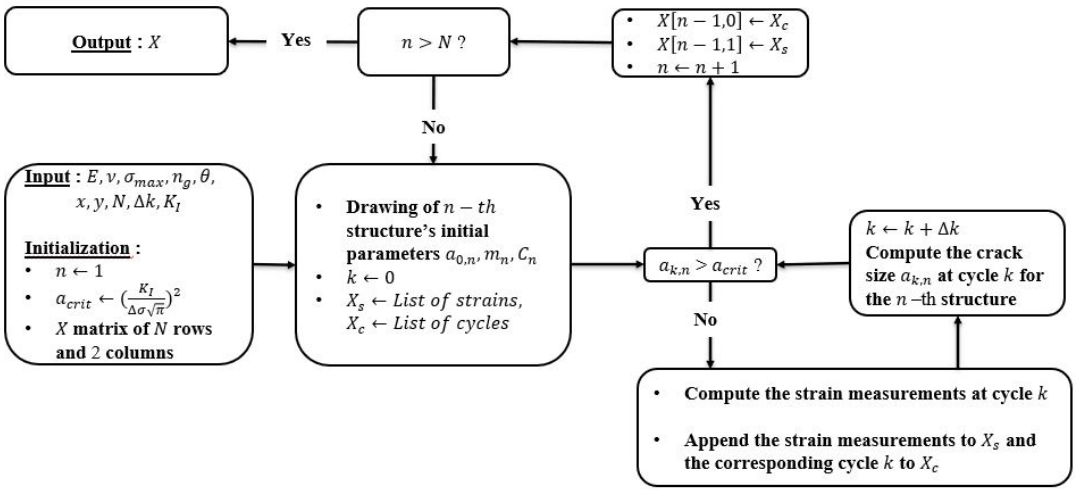

Fig. 3. Flow chart of the strain gauge sequence generation.

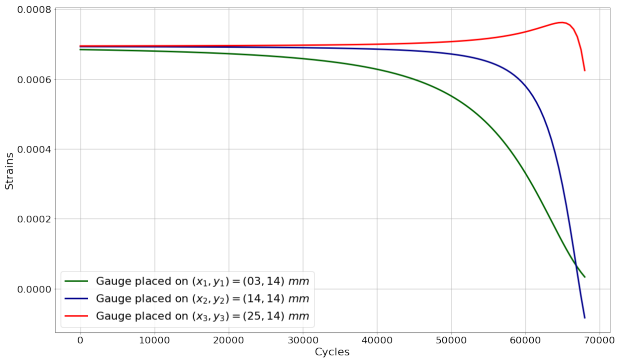

Fig. 4. Strain values time series corresponding to a sensor sample generated. In this illustration, three strain gauges are placed at the following positions (x,y) : (3,14), (14,14) and (25,14) mm.

be seen representative of real experiments under fatigue loading where the strain state is monitored at three strain gauge positions. The strain data, or measurement sequences, are obtained until the critical crack size a_{crit} is reached. In this experiment, an Aluminum alloy 7075-T6 plate was considered, which is typical of aeronautic structures. The elastic parameters considered are Young's modulus $E = 71.7GPa$ and Poisson's ratio $\nu = 0.33$ (assumed to be constant at their nominal values). The critical crack size a_{crit} that causes structural failure can be calculated by the empirical formula in Eq. (3), in which $K_I = 19,7\ MPa\sqrt{m}$ and $\Delta\sigma = \sigma_{max} - \sigma_{min} = 78,6\ MPa$ with $\sigma_{min} = 0\ MPa$ in this work.

In this series of numerical experiments, it is assumed that the initial crack position is at the origin of the (x, y) reference frame and that the crack is along the x direction. Furthermore, it is assumed that the strain gauges are placed at an angle θ = 45° with x-axis, so they are sensitive to both ϵ_{11} and ϵ_{22} strains. The $n_g = 3$ strain gauges are placed at the following positions (x,y) : (3,14), (14,14) and (25,14) mm. Considering that the evolution of the changes from one cycle to another are small, it was decided to collect the data every $\Delta k = 500$ cycles. Fig. 4 illustrates an instance of sequences generated for a single structure until failure.

In the following illustration, artificial experiments are set up in order to acquire a training, validation and testing data set. The validation set is used to monitor and optimize the models hyperparameters during the training phase; the testing set is used to evaluate the performance of the trained models as a held-out data set that has not been used prior, either for training the model or tuning the model hyperparameters. For training, data sets of various sizes $N_{T_1} = 100$, $N_{T_2} = 500$, $N_{T_3} = 1000$ structures are generated, while for validation a set of $N_V = 100$ structures is generated, and for testing, a set of $N_{Test} = 100$ structures is generated. Each structure n is defined by three parameters that vary from structure to structure within the data set (i.e. the initial crack length $a_{0,n}$, C_n and the exponent m_n), such that :

1) a_0 varies following a Gaussian law with a mean of $1 \cdot 10^{-3}[m]$ and a coefficient of variation (C_v) of 0.125, such that $C_v(a_0) = \frac{\sigma_{a_0}}{\mu_{a_0}}$, where the Gaussian is truncated to avoid negative values.
2) C is distributed following a log-normal law with a geometric mean equal to $1 \cdot 10^{-10}$ and a geometric standard deviation such that the exponential of the upper and lower bounds of the 95% confidence interval for $\log C$ have a ratio of $8 \cdot 10^3$, which is consistent with the data from [25].
3) Finally, m is distributed following a Gaussian law such that the bounds of the 95% confidence interval are 2 and 4, as suggests [18] and is again roughly consistent with [25]. Again, the distribution is truncated at the lower end at 0 so that m cannot become negative.

Parameter	Denotation	Type	Value	Unit
Elastic parameters				
Young's modulus	E	Deterministic	71.7	GPa
Poisson's ratio	ν	Deterministic	0.33	-
Strain field parameters				
Maximum stress intensity	σ_{max}	Deterministic	$78, 6.10^6$	Pa
Fracture toughness	K_I	Deterministic	$19, 7.10^6$	$Pa\sqrt{m}$
Strain gauges				
Number of gauges placed	n_g	Deterministic	3	-
Position of the gauges placed	$(x_i, y_i)_{i=1,..,n_g}$	Deterministic	$(3, 14), (14, 14), (25, 14)$	mm
Angle of the gauges placed	θ	Deterministic	45	deg
Initialization parameters				
Initial crack size	a_0	Gaussian distribution	$\mathcal{N}(\mu_{a_0}, \sigma_{a_0})$	m
Mean of a_0	μ_{a_0}	Deterministic	1.10^{-3}	m
Standard deviation of a_0	σ_{a_0}	Deterministic	$0, 125.10^{-3}$	m
Paris-Erdogan's law parameters	$(m, \log C)$	Multivariate Gaussian distribution	$\mathcal{N}(\mu_m, \sigma_m, \mu_{\log C}, \sigma_{\log C}, \rho)$	-
Mean of m	μ_m	Deterministic	$3, 5$	-
Standard deviation of m	σ_m	Deterministic	$0, 125$	-
Mean of C	μ_C	Deterministic	1.10^{-10}	-
Standard deviation of C	σ_C	Deterministic	5.10^{-11}	-
Correlation coefficient of m and $\log C$	ρ	Deterministic	-0.996	-
Generated data set				
Number of training structures	$(N_{T_1}, N_{T_2}, N_{T_3})$	Deterministic	$(100, 500, 1000)$	-
Number of validation structures	N_V	Deterministic	100	-
Number of testing structures	N_{test}	Deterministic	100	-
Data collection interval	Δk	Deterministic	500	-

TABLE I
PARAMETERS FOR NUMERICAL STUDY.

For the material studied in this paper (i.e. alluminum alloys), m and $\log C$ are assumed to follow a multivariate distribution with a negative correlation coefficient of $\rho = -0.996$ [26]. The parameters of this numerical case study are summarized in Table I.

B. Methodology

In this section, an overview of some Deep Learning methods commonly used in prognostics is provided, followed by a description of the problem considered in this paper and the way these methods were implemented on it.

1) Deep Learning in Prognostics: Deep Learning has shown multiple times to provide good results when applied to RUL prediction [9], [27]. Among the deep learning techniques, we can enumerate two widely used algorithms in the prognostics field :

- Recurrent Neural Networks
- Convolutional Neural Networks

Both will be briefly introduced below.

a) Recurrent Neural Networks: The most common type of Deep Learning model for Time Series Forecasting are Recurrent Neural Networks (RNN) [28]. The algorithm remembers its input due to an internal memory, which makes it well suited for problems that involve sequentially evolving data. Good results have been obtained by applying RNNs to a variety of problems in non-PHM fields, such as speech recognition or language modeling [29]–[31]. Due to their ability to capture time-dependent relationships, RNNs have achieved interest among PHM researchers as well, especially given the sequential nature of the sensor data in the prognostics field (e.g. sensors data).

However, standard RNNs are limited to looking back only a few steps due to the vanishing gradient or exploding gradient problem (see for example [32], [33]). To address this issue, Long Short-Term Memory (LSTM) networks were proposed, and have established themselves as one of the most used Deep Learning model types in many fields and especially in Natural Language Processing (NLP) [34]. More recently, LSTM networks have also grown in popularity and have been used by several researchers in the PHM community [9]. One of the major drawbacks of LSTM concerns their relatively high computational cost and memory requirement for training [35]. A slightly simplified variation of the LSTM is the Gated Recurrent Unit (GRU), introduced by Cho et al. [36], gaining in popularity in recent years due to its relative simplicity [37]. In [38], results showed GRU model could achieve competitive results with a better training performance than LSTM for RUL estimation.

b) Convolutional Neural Networks: Convolutional Neural Networks (CNN) concern a specific type of deep neural network inspired by the organization of neurons in the visual cortex. Presented by LeCun et al. in 1998 [39], CNNs achieved significant success in many research and industry fields including computer vision, natural language processing and speech recognition [40], [41]. Input data for CNNs are usually 2-dimensional (e.g. height and width pixels for images) so to learn abstract spatial features. Li et al. [42] investigated how a 2D-DeepCNN model can be used in prognostics and remaining useful life prediction. 1D-CNNs have also been introduced to the analysis of time

sequences in RUL estimation. The key difference between 1D-CNN, 2D-CNN and 3D-CNN is the dimensionality and management of the input data and how the feature detector (or filter) slides across the data. In this paper, only 1D-CNNs will be considered due to the format of our data sets (Time Series). Indeed, the application of the CNN architecture to time series prediction aims to exploit the filters' feature extraction capability demonstrated in image classification, and CNNs are easier to train than recurrent neural networks due to the implementation of convolutional rather than recursive operations, allowing improved numerical efficiency. However, note that CNNs were initially introduced as a classifiers [39], thus more suitable for classification problems than regression problems in sequence modeling (i.e. for RUL estimation problems that have been essentially considered as regression problems so far).

According to [43], sequence modeling was synonymous with recurrent networks for most deep learning practitioners until 2018. Bai et al. [43] introduced a novel CNN architecture for sequence modeling, using dilated causal convolution to preserve the causal order of the input time series : Temporal Causal Networks (TCN). Their results showed that the TCN outperformed the standard recurrent neural networks (RNN, LSTM and GRU) in most Sequence Modeling Tasks (better performances on ten tasks out of eleven). Among PHM researchers, Liu et al. [44] proposed a TCN model for RUL prediction of rolling bearings based on raw vibration data. In their paper, results confirmed that the proposed model is able to outperform generic recurrent architectures such as LSTM and GRU in sequence modeling. Moreover, the authors showed that offline training of the proposed model is almost four times faster than an LSTM network. Indeed, according to Bai et al. [43], the TCN has several advantages that make it superior to the Recurrent Neural Networks (RNN, LSTM, GRU), specifically :

- better control of the model's memory size with a flexible receptive field size (stacking more dilated causal convolutional layers, using larger dilation factors, or increasing the filter size);
- a backpropagation path different from the one used by recurrent neural networks, which allows to avoid the exploding/vanishing problem.

2) Problem statement: In this paper, the RUL prediction problem is considered as a multivariate time-series-related regression problem and, as illustration, the five most commonly used deep learning-based algorithms in Time Series Forecasting (RNN, LSTM, GRU, 1D-CNN and TCN) have been applied to the generated data set.

In the training phase, a sliding window approach is used. At each time-step t, the input of the predictive model correspond to the current and past measures, such that $X_t := (x_{t-n_w+1}, \ldots, x_t) \in \mathbb{R}^{n_g \text{ x } n_w}$ where n_g is the number of strain gauges and thus of time series and n_w is the length of the sliding window; the output of the predictive model is a point-wise estimation of the RUL of the structure at time t, denoted $\hat{RUL}_t \in \mathbb{R}$. Note that each training structure is composed of S samples (sliding windows of size n_w) with $S = n_T - n_w$, n_T the number of timesteps in the sequence.

After training, the models are evaluated on the testing set composed of N_{test} different structures, and for each structure a unique RUL estimation is performed at a time t_n^*. For each structure $n \in \{1, \ldots, N_{test}\}$, the parameter t_n^* is randomly drawn such that $t_n^* \sim k_{crit}^n \times \mathcal{U}([0,33; 0,9])$, where k_{crit}^n is the cycle of failure for the n-th structure. In plain words, this means that we carry out a test prediction for the RUL at a time t_n^* which is drawn uniformly between 33% and 90% of the sequence's length. Hence, the input data for the model is $X_{t_n^*} = (x_{t_n^*-n_w+1}, \ldots, x_{t_n^*}) \in \mathbb{R}^{n_g \text{ x } n_w}$ and the output of the model is $\hat{RUL}_{t_n^*} \in \mathbb{R}$.

3) Training and Evaluation: For the recurrent neural networks, the input data is pre-processed using a min-max scaler before being fed into the models. Fig. 5 shows an example of how the time series data of the strain gauges from a single structure are processed, and gives an illustration of recurrent neural networks architecture used in this work.

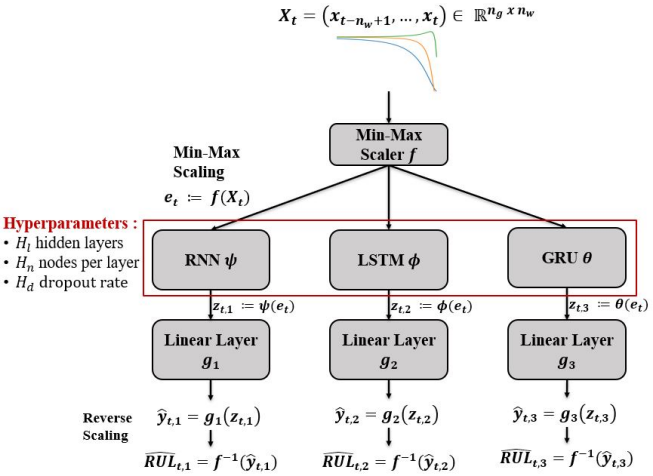

Fig. 5. Brief illustration of recurrent neural networks (RNN, LSTM and GRU) architectures. n_g corresponds the number of gauges placed (i.e. number of time series), and n_w to the window size.

The first layer of the 1D-CNN's architecture is a normalization layer. The architecture of the 1D-CNN models is illustrated in Fig. 6.

The input data is not pre-processed for the TCN model and we have kept the same architecture as presented in [45], illustrated in Fig. 7.

Given that the RUL estimation problem is considered as a regression problem, we aim to minimize a mean squared error loss L_{MSE} during the training phase, and the mean absolute percentage error (*MAPE*) metric is used to evaluate the performance of the investigated models such that :

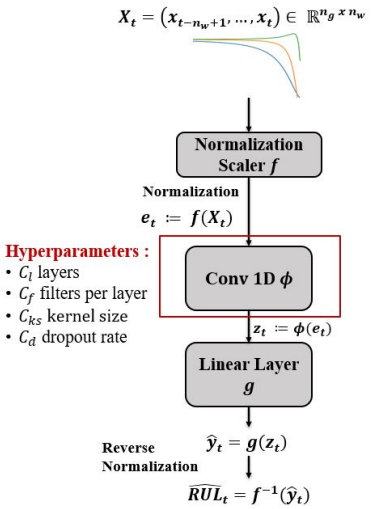

Fig. 6. Brief illustration of convolutions neural networks (1D-CNN) architecture. n_g corresponds the number of gauges placed (i.e. number of time series), and n_w to the window size.

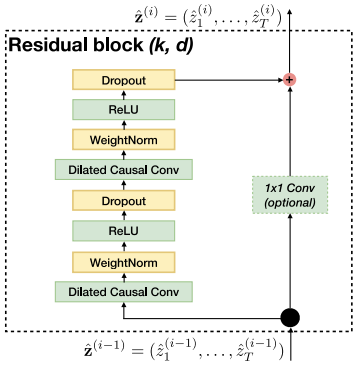

Fig. 7. Architecture of a residual block of TCNs introduced in [45]. A TCN is a stack of k residual blocks, each composed of two 1D-CNN layers (same hyperparameters as illustrated in Fig. 6) with a dilation factor d followed by a weight normalization layer used for regularization [46].

$$L_{MSE} = \frac{1}{S}\sum_{i=1}^{S}(RUL_i - \hat{RUL}_i)^2 \quad (16)$$

$$MAPE = \frac{1}{S}\sum_{i=1}^{S}\left|\frac{RUL_i - \hat{RUL}_i}{RUL_i}\right| * 100 \quad (17)$$

where S is the number of samples with $\hat{RUL}_.$ being the prediction and $RUL_.$ the target value.

In the training phase, our strategy consists of a hyperparameters optimization stage using a Grid Search algorithm [47], followed by a Fine-tuning stage. During the Fine-tuning stage, we use the Adam optimizer with default parameters and we decrease the learning rate incrementally : we sequentially try the learning rates $10^{-3}, 10^{-4}, 10^{-5}$ for a predefined number of epochs, saving the model weights each time the validation MAPE decreases; the weights of the best model are loaded each time the learning rate is lowered. By using this approach, the model in the early stages of training with a high learning rate is less likely to get stuck in a local minimum and explores a wider range of possible configurations. As the training comes closer to an optimum, the decaying learning rate helps with convergence and avoiding oscillations around local minima [48].

C. Experiments and Results

Due to the time series nature of the data and the sliding window procedure (with $n_w = 30$ after some preliminary experiments), the training sets with $N_{T_1} = 100$, $N_{T_2} = 500$ and $N_{T_3} = 1000$ structures correspond to respectively 10956, 54365 and 108452 training samples. The validation set is processed in the same way : N_V contains 100 validation structures thus 10937 samples. For testing, a set of $N_{Test} = 100$ structures is used, hence composed of 100 samples since only one RUL estimation is performed per structure.

For the recurrent neural networks, the optimization strategy was applied on 100, 500 and 1000 structures. The optimal hyperparameters found are summarized in Table II.

Training structures	100			500			1000		
Model	RNN	LSTM	GRU	RNN	LSTM	GRU	RNN	LSTM	GRU
Hidden layers	3	2	3	3	2	2	3	2	2
Nodes per layer	32	64	256	32	128	256	32	128	256
Dropout rate	0	0	0	0	0	0	0	0	0

TABLE II
BEST MODELS OF THE RNN, LSTM AND GRU HYPERPARAMETER OPTIMIZATION FOR 100, 500 AND 1000 TRAINING STRUCTURES

For the 1D-CNN and TCN model the hyperparameter optimization were again performed on 100, 500 and 1000 training structures, and the optimal hyperparameters found are summarized in Table III. Note that the dilation factor is set to $d = 6$, so that the receptive field is of size $2^{d-1} = 2^5 = 32$, thus able to capture relationships over a time sequence of size $n_w = 30$ in this work, as described in [45].

Training structures	100	500	1000
Convolutional layers	6	2	4
Filters per layer	20	40	25
Kernel size	6	12	9
Dropout rate	0.0	0.0	0.0

(a) Best models of the 1D-CNN.

Training structures	100	500	1000
TCN residual blocks	8	8	8
Filters per layer	30	25	35
Kernel size	2	2	2
Dropout rate	0.0	0.0	0.0

(b) Best models of the TCN.

TABLE III
HYPERPARAMETER OPTIMIZATION FOR 100, 500 AND 1000 TRAINING STRUCTURES.

As the hyperparameters are optimized, the resulting models are fine-tuned on the same data set with an adaptative learning rate to optimize their performance, and then evaluated on 100 structures of the testing data set; for each structure we only have one sample to evaluate, thus the models are evaluated on 100 samples. Table IV shows an overview of the achieved accuracy for all recurrent neural networks (RNN, LSTM and GRU) and convolutional neural networks (1D-CNN and TCN).

Training structures	100		500		1000	
$MAPE(\%)$	Val	Test	Val	Test	Val	Test
RNN	3.5	2.95	2.22	1.54	0.91	0.67
LSTM	**0.55**	**0.65**	0.25	1.15	1.50	1.24
GRU	1.72	1.22	**0.05**	**0.02**	**0.08**	**0.04**
1D-CNN	4.19	3.13	1.06	0.82	0.58	0.54
TCN	2.57	2.35	0.76	0.66	0.77	0.69

TABLE IV
REGRESSION TASK : MAPE (%) OF ALL FINE TUNED RNN (RNN, LSTM AND GRU) AND CNN MODELS (1D-CNN AND TCN) ON THE VALIDATION AND TESTING DATA SETS

LSTMs outperformed the other algorithms when trained on 100 structures (0.65% testing MAPE score), while the best performance in general was achieved by GRU models (0.02% testing MAPE score when trained on 500 structures, and 0.04% testing MAPE score when trained on 1000 structures). Indeed, recurrent neural networks (especially GRU) appear to be the best suited for the regression task in this RUL estimation problem. Nevertheless, we can see that more available training data leads to a better performance of the models and a better identification of the most suitable models for the problem.

IV. CONCLUSION

In this paper, a framework and code for synthetically generating high dimensional data sets for a realistic fatigue damage prognostics problem has been presented. The proposed framework generates multivariate run-to-failure time series data for structures subject to fatigue loading, consisting of synthetic mechanical strain data sets (i.e. synthetic strain gauges) and associated RUL based on the Paris-Erdogan crack growth model. The authors believe that the proposed framework will help facilitate the benchmarking of latest ML algorithms for fatigue damage prognostics applications, in particular in the aerospace domain (e.g. fuselage panels).

As illustration, pre-cracked Aluminum alloy 7075-T6 plates were considered, which are typical of aeronautic structures, and the applicability of some of the most commonly used DL models to address failure prognostics (including RNN, LSTM, GRU, 1D-CNN, and TCN) have been studied. Without the "flaws" of real world data, good results were expected and obtained. Indeed, there is no noise in the data which would not be the case in reality : a gaussian noise can be added to time series in order to mimic the effect of aleatoric uncertainties that occur in real world applications. In next steps, building upon the possibility to "play" with the initialized parameters, this illustration case study can be further varied in order to complexify the data set and make it as realistic as possible.

Furthermore, the authors believe that the use of synthetic data can improve the prognostic performance of data-driven models in real-world cases in the absence of training data or with very limited labeled data, using knowledge transfer when a model is pre-trained on a large set of related synthetic data (e.g. Transfer Learning [49], [50]), and the presented framework might be useful in this aspect as well.

ACKNOWLEDGMENT

This work was partially funded by the French "Occitanie Region" under the Predict project. This funding is gratefully acknowledged. This work has been carried out on the supercomputers PANDO (ISAE-SUPAERO, Toulouse) and Olympe (CALMIP, Toulouse, project n°21042). Authors are grateful to ISAE-SUPAERO and CALMIP for the hours allocated to this project. The authors would like to thank Christian Thomas Nitschke for his initial input on modelling the crack growth problem.

REFERENCES

[1] E. Zio, "Prognostics and health management of industrial equipment," in *Diagnostics and prognostics of engineering systems: methods and techniques*. IGI Global, 2013, pp. 333–356.
[2] A. Vasudevan, K. Sadananda, and N. Iyyer, "Fatigue damage analysis: Issues and challenges," *International Journal of Fatigue*, vol. 82, pp. 120–133, 2016.
[3] 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.
[4] K. Javed, "A robust & reliable data-driven prognostics approach based on extreme learning machine and fuzzy clustering." Ph.D. dissertation, 2014.
[5] K. L. Tsui, N. Chen, Q. Zhou, Y. Hai, and W. Wang, "Prognostics and health management: A review on data driven approaches," *Mathematical Problems in Engineering*, vol. 2015, 2015.
[6] 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.
[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," vol. 521, no. 7553, pp. 436–444.
[8] A. Voulodimos, N. Doulamis, G. Bebis, and T. Stathaki, "Recent developments in deep learning for engineering applications," vol. 2018, p. 8141259.
[9] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, "Potential, challenges and future directions for deep learning in prognostics and health management applications," *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103678, 2020.
[10] A. Theissler, J. Pérez-Velázquez, M. Kettelgerdes, and G. Elger, "Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry," *Reliability engineering & system safety*, vol. 215, p. 107864, 2021.
[11] P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Chebel-Morello, N. Zerhouni, and C. Varnier, "Pronostia: An experimental platform for bearings accelerated degradation tests." in *IEEE International Conference on Prognostics and Health Management, PHM'12*. IEEE Catalog Number: CPF12PHM-CDR, 2012, pp. 1–8.
[12] B. Saha and K. Goebel, "Battery data set," *NASA AMES prognostics data repository*, 2007.

[13] O. F. Eker, F. Camci, and I. K. Jennions, "Major challenges in prognostics: Study on benchmarking prognostics datasets," in *PHM Society European Conference*, vol. 1, no. 1, 2012.

[14] D. A. Virkler, B. Hillberry, and P. Goel, "The statistical nature of fatigue crack propagation," 1979.

[15] A. Ray and S. Tangirala, "Stochastic modeling of fatigue crack dynamics for on-line failure prognostics," *IEEE Transactions on Control Systems Technology*, vol. 4, no. 4, pp. 443–451, 1996.

[16] A. Abbasi, F. Nazari, and C. Nataraj, "Application of long short-term memory neural network to crack propagation prognostics," in *2020 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2020, pp. 1–6.

[17] P. Paris and F. Erdogan, "A critical analysis of crack propagation laws," vol. 85, no. 4, pp. 528–533.

[18] A. T. Zehnder, *Fracture mechanics*, ser. Lecture notes in applied and computational mechanics. London ; New York: Springer Science+Business Media, 2012, no. 62, oCLC: ocn755698387.

[19] Y. Wang, C. Gogu, N. H. Kim, R. T. Haftka, N. Binaud, and C. Bes, "Noise-dependent ranking of prognostics algorithms based on discrepancy without true damage information," *Reliability Engineering & System Safety*, Oct. 2017.

[20] C.-T. Sun and Z. Jin, "The elastic stress field around a crack tip," *Fracture Mechanics*, pp. 25–75, 2012.

[21] A. T. Zehnder and M. J. Viz, "Fracture mechanics of thin plates and shells under combined membrane, bending, and twisting loads," *Appl. Mech. Rev.*, vol. 58, no. 1, pp. 37–48, 2005.

[22] A. F. Bower, *Applied mechanics of solids*. Boca Raton: CRC Press, 2010, oCLC: ocn277196164. [Online]. Available: solidmechanics.org

[23] J. Benson and D. Edmonds, "Relationship between the parameters c and m of paris' law for fatigue crack growth in a low-alloy steel," *Scr. Metall.;(United States)*, vol. 12, no. 7, 1978.

[24] M. Cortie and G. Garrett, "On the correlation between the c and m in the paris equation for fatigue crack propagation," *Engineering fracture mechanics*, vol. 30, no. 1, pp. 49–58, 1988.

[25] G. Sinclair and R. Pieri, "On obtaining fatigue crack growth parameters from the literature," *International Journal of Fatigue*, vol. 12, no. 1, pp. 57–62, Jan. 1990. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/014211239090343D

[26] E. H. Niccolls, "A correlation for fatigue crack growth rate," *Scripta Metallurgica*, vol. 10, no. 4, pp. 295–298, Apr. 1976. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/003697487690079X

[27] J. J. Montero Jimenez, S. Schwartz, R. Vingerhoeds, B. Grabot, and M. Salaün, "Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics," vol. 56, pp. 539–557. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612520301187

[28] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *International Journal of Forecasting*, vol. 37, no. 1, pp. 388–427, 2021.

[29] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," 2014.

[30] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[31] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *INTERSPEECH*, 2013.

[32] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[33] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.

[34] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation."

[35] T. Masuko, "Computational cost reduction of long short-term memory based on simultaneous compression of input and hidden state," in *2017 IEEE automatic speech recognition and understanding workshop (ASRU)*. IEEE, 2017, pp. 126–133.

[36] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[37] R. Rana, "Gated recurrent unit (gru) for emotion classification from noisy speech," *arXiv preprint arXiv:1612.07778*, 2016.

[38] M. Baptista, H. Prendinger, and E. Henriques, "Prognostics in aeronautics with deep recurrent neural networks," in *PHM Society European Conference*, vol. 5, no. 1, 2020, pp. 11–11.

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[40] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *arXiv preprint arXiv:1901.06032*, 2019.

[41] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *arXiv preprint arXiv:1905.03554*, 2019.

[42] 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.

[43] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[44] C. Liu, L. Zhang, and C. Wu, "Direct remaining useful life prediction for rolling bearing using temporal convolutional networks," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2019, pp. 2965–2971.

[45] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," vol. abs/1803.01271.

[46] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[47] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning*. Springer, Cham, pp. 3–33.

[48] K. You, M. Long, J. Wang, and M. I. Jordan, "How does learning rate decay help modern neural networks?"

[49] J. C. Paetzold, O. Schoppe, R. Al-Maskari, G. Tetteh, V. Efremov, M. I. Todorov, R. Cai, H. Mai, Z. Rong, A. Ertuerk *et al.*, "Transfer learning from synthetic data reduces need for labels to segment brain vasculature and neural pathways in 3d," in *International Conference on Medical Imaging with Deep Learning–Extended Abstract Track*, 2019.

[50] C. Douarre, R. Schielein, C. Frindel, S. Gerth, and D. Rousseau, "Transfer learning from synthetic data applied to soil–root segmentation in x-ray tomography images," *Journal of Imaging*, vol. 4, no. 5, p. 65, 2018.