Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 51 (2020) 1207–1214

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021)
15-18 June 2021, Athens, Greece.

On Making Factories Smarter through Actionable Predictions based on Time-Series Data

Sophia Karagiorgou[a,*], Christos Rountos[a], Georgia Chatzimarkaki[a], Georgios Vafeiadis[a], Dimitrios Ntalaperas[a], Danae Vergeti[a], Dimitrios Alexandrou[a]

[a] *UBITECH LTD, Thessalias 8 and Etolias 10, Chalandri, P.C. 15231, Greece*

Abstract

The emergence of the Industrial Internet of Things paves the way for enhancing the real-time monitoring capabilities of contemporary factories. This can be materialised through sensors collecting data which can be further analysed. This paradigm enables the detection of indicators concerning systems' degradation and facilitates corrective actions to be performed ahead of time. In this paper, we propose a damage prediction framework exploiting data that are coming from the core IT system in a steel industry use case. The novelty of our approach lies in the exploitation of Deep Learning techniques over streaming operational sensor data. To evaluate the framework, real-life data are collected and analysed based on daily operational activities enriched with the Remaining Useful Life (RUL). The Remaining Useful Life is automatically computed. We also periodically align the damage attribute, which is recorded in the planned activities and is performed by the maintenance engineers. We demonstrate the framework's potential by conducting a quantitative and qualitative analysis to timely identify rare events based on historical data and predict dynamic behavioural changes in the manufacturing settings.

Keywords: Industry 4.0; Industrial Internet of Things Architectures; Predictive Analytics for Production Monitoring; Advanced Machine Learning and Artificial Intelligence Approaches for Industrial Use Cases

1. Introduction

Predictive Maintenance (PM) is becoming a kind of hype these days and is being proclaimed as the 'killer app' for the Industrial Internet of Things (IIoT) and Factories of the Future (FoF). The shift from compartmentalized technical infrastructures to digitized ecosystems along with the rise of IIoT technologies influence the current approaches by introducing a plethora of easily interconnected sensors collecting real-time machinery data capturing the temperature, acceleration, vibration, greasing or lube oil levels of moving parts and operational conditions from monitored assets. A key differentiator in real-time IIoT monitoring is that it allows for different assets and systems to connect, work together, share and analyse data which can be further translated into actions. In the course of predictive maintenance operations, the ultimate goal is to monitor through the sensors the performance and condition of equipment during normal operation in order to reduce the likelihood of damages or failures. Provided that a set of functionalities first enable to predict when equipment failure may occur based on certain factors, then a set of decision making actions occurs followed by preventing the failure through regularly corrective and scheduled maintenance. There are numerous examples that have showcased the impact of a disruption from a failure in terms of time, cost and public safety. For instance, in March 2008 a failure in the baggage system at Heathrow airport is estimated to have cost $32 million and affected 140,000 people [3].

Today's democratized Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning (DL) enable the computation of predictive analytics. System's health through continuous monitoring and analysis is of great importance in industrial informatics field. Their combination with IIoT help a growing number of manufacturers to monitor their production line in real-time and prevent failures before they occur. A good solution to this problem is to use continuous monitoring and advanced analytics to detect or predict the status of a manufacturing damage.

Converging IIoT with AI and advanced ML holds the potential to realize the digitized ecosystems, the industrial trans-

* Corresponding author. Tel.: +30 216 5000 530 ; fax: +30 216 5000 599.
E-mail address: skaragiorgou@ubitech.eu (Sophia Karagiorgou).

2351-9789 © 2020 The Authors. Published by Elsevier Ltd.

10.1016/j.promfg.2020.10.169

formation known as Industry 4.0 and the concept of smart (self)-optimizing digital infrastructures empowered by Predictive Maintenance processes. This approach can further drive the proactive execution of decisions, which can boost productivity, achieve better regulation compliance and workplace safety, ensure an eco-friendly work environment with low energy consumption, high environmental preservation and overall costs reduction. At the same time, this convergence seeks to define the golden section to timely do work on an asset, so that maintenance frequency is as low as possible and reliability is as high as possible without unnecessary interruptions by saving operational time, maximising resources and cutting costs. In the short term, the Predictive Maintenance methodology is expected to be streamlined as a proactive, efficient, automated, and high performing approach.

However, despite the acknowledgement for direct and indirect benefits from this methodology, still it is not well perceived by the industry, mainly due to the *lack of services and tools* at a reasonable cost level, the *required amount of labeled data*, the *high requirements in technical and field expertise* and the *scepticism about its efficiency*. In many cases, bottlenecks are also introduced by the data heterogeneity, the lack of automated data acquisition processes and the inefficacy of the traditional statistical and machine learning approaches [6], [7]. The latter is also affected by the inherent difficulties in extracting significant features which are indicating the differentiation of the monitored asset until failure and in revealing the high-dimensional attributes of massive signals, which affect the accuracy and efficiency of health assessment of the system. In fact, these are the main reasons why there is a shortcoming to implement a “preferred” system to the degree it is needed.

This paper proposes an approach which bridges the gap that currently maintenance is realized via deterministic, frequently scheduled and costly machinery replacements to proactively prevent breakdowns as well as the high requirements in technical and field expertise arising from these complex systems. This is currently met due to the heterogeneity of the existing solutions (e.g. different sensor types, diverse collected data, the lack of semantic alignment, need for prior knowledge of the technical infrastructure, the organisation’s policies and needs, etc.) and their practical implementation [11], [12]. Also, the limited availability of labeled data, as well as the complex nature of multivariate time-series produced by sensors (i.e. communication errors, connection timeouts, need for semantic alignment, etc.) is a challenging task, ultimately aiming to efficiently and timely predict undesirable situations. As this paper will show in detail, the major factors influencing the availability, provenance and management of the collected data have little to do with the quality of the predictions produced out of them. Therefore, the contributions of this paper are as follows:

- Develop novel cross-platform standardised connectors for all key information sources along with filtering, faceting and semantic concepts over the data collected in real-time, which are periodically aligned and enriched with recorded maintenance activities;
- Propose a set of Deep Learning algorithms for Predictive Maintenance, by exploiting state of the art training models for time-series regression and classification, such as Long Short Term Memory (LSTM), which has been proven suitable for sequence modeling and temporal dependencies learning;
- Evaluate the proposed framework with real-time sensor data and validate its effectiveness through the performance metrics of mean squared error, precision, recall and accuracy.

2. Related Work

Sensor interface devices are considered to be highly essential for sensor data collection through industrial Wireless Sensor Networks (WSN) in IIoT environments. Due to this, a variety of new methods towards the demonstration of smart sensor interfaces are proposed. A similar method introduced by Chi et al. [5] focuses on the design of a re-configurable smart sensor interface for industrial WSN in IIoT environments, in which a Complex Programmable Logic Device (CPLD) is adopted as the core controller. Thus, it can read data in parallel and in real-time with high speed on multiple different sensor interfaces [16]. Furthermore, sensor data collection may comprise sensor nodes where sensed data are collected and subsequently forwarded to a central base station for further processing as introduced by Wang et al. [27]. The latter is considered as a typical approach, and is similar to the methodology that we followed, in which the collected data are stored in order to be further analysed.

Detection of technical malfunctions is determined by a diagnosis and prognosis system where input data from various sources are received, appropriate computational processing is executed towards abnormalities detection and removal (i.e. data cleaning, outliers, missing values elimination, etc.) [24] and decision options are produced. Compared to top-down modeling provided by the traditional physics-based models [31], [33], data-driven machine health monitoring systems offer a new paradigm of bottom-up solution for the detection of faults and unexpected damages after the occurrence of certain failures (i.e. diagnosis), the prediction of future working conditions and forecasts about the Remaining Useful Life (i.e. prognosis) [4], [8], [17]. However, the complex and noisy working condition hinders the construction of physical models, which make the modelling of complex dynamic systems very difficult [23]. Most of these physics-based models are unable to be updated with *on-line measured data*, which *limits their effectiveness and flexibility*.

A plethora of simple or complex algorithms with different outputs is required, also feeding later processing stages, by implementing a properly configured timed or event-triggered chain-able execution of methods with parameterized instances. The challenge comes when translating the ambiguous methods results from complex multi-modal technical systems towards the status evaluation of the entire system. Regression, classification [9] and/or survival models facilitate to predict failure over time [22] or the Remaining Useful Life (RUL) [28]. Also, anomalous operations [32] can be efficiently flagged [21] by data records isolation in linear time [2].

Besides, Deep Learning is recently used for training deep neural models for Predictive Maintenance of monitored assets [29], based on the streaming and/or batch nature of data coming from the available sensors. Temperature, acceleration and vibration measurements, lube oil levels, machinery utilization, failure events over time, etc., due to their sequential nature can be modelled with neural network models such as the Long Short Term Memory (LSTM) neural network, which is capable to learn the past dependencies over the data that may influence future events. It has been shown that such models can achieve high performance and precision in health assessment and Predictive Maintenance tasks in a variety of industries [13], [18], [20] without requiring system-specific engineered features depending on high level experts' experience [16], [26].

In the context of predictive algorithms and their selection for a particular industrial business case, another approach introduced by Sikorska et al. [25] considers appropriate model selection for successful practical implementation by taking into account not only a mathematical understanding of each model type, but also the assessment of how a particular industrial organization intends to utilize the model and its outputs. Adhikari et al. [1] dealt with time-series prediction and classification methods and proposed a linear combination method for time-series forecasting that determines the combining weights through a novel neural network structure. The designed neural network successively recognized the weight patterns of the constituent models from their past forecasting records and then predicted the desired set of the combining weights.

Although the current approaches of data-driven failure prediction systems use monitoring devices, methods for detection of technical malfunctions, time-series prediction and classification methods, each individual work typically focuses either on predictive algorithms or on costly sensor interfaces [30] without unleashing the full potential of their combination. In general, the problem of failure prediction is much more complex, particularly as the problem of data-driven failure prediction is more complex.

Compared to the aforementioned approaches, the proposed framework differs by introducing a data-driven prediction system, which exploits Deep Learning techniques from streaming operational sensor data which are categorized by filters in order to facilitate the process of detection and are simultaneously stored in a centralized storage infrastructure in real-time. The streaming data are analysed to provide real-time predictions for each functional asset monitored through sensors in the production line, which subsequently can trigger potential reactive measures for timely preventing the respective failures. A more significant aspect of the proposed framework is its ability to manage data at scale, in order to facilitate the time and complexity reduction in the required analysis.

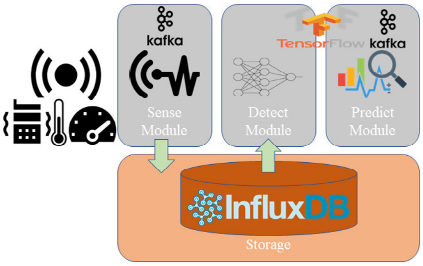

Fig. 1: High level architecture.

3. A Deep Analysis Framework exploiting Industrial Time-Series Streaming Data

The Predictive Maintenance (PM) framework introduced in this paper consists of 3 core components: the *Sense Module*, the *Detect Module* and the *Predict Module*. In Figure 1, we present the high level architecture of the proposed framework.

3.1. Deep Analysis Methodology

Long Short-Term Memory Networks (LSTM) [14] is a Recurrent Neural Network (RNN) architecture, which can be viewed as a deep neural network rolled in time. The internal state of the network does not only rely on the current input but it also depends on the states of the system during previous times. This is fundamentally different from the feed-forward neural networks, since LSTM network is designed to capture information of sequential data and maintain contextual features from past observation, in order to predict future ones. This neural architecture has been successfully applied in many real-world problems involving sequential data, mainly in the field of Natural Language Processing (NLP) [10]. In this way, LSTM has been proven a precise and efficient approach of modeling our world of sequential information over time in a simple way.

Given the strong temporal dependencies in the industrial system failure prediction, in this work we explore the application of LSTM neural networks to model the dynamic nature of manufacturing systems. A step towards this goal can be achieved by forecasting the time-series of sensor's measurements, which can contain important indicators of upcoming system failures or abnormal behaviors. This is the aim of the present study, where we investigate the accuracy of the prediction models on a generalised data-driven basis (cf. Algorithm 1).

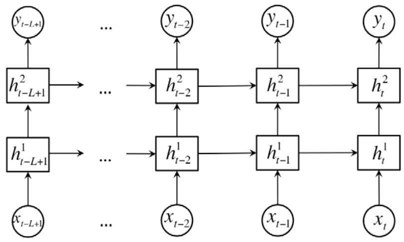

Fig. 2: A many-to-many deep recurrent neural network forecasting architecture.

Figure 2 presents the forecasting architecture of the recurrent neural network we used in this paper. The rectangles represent the hidden layers, and the circles at the bottom and on the top represent the input layer and output layer, separately. The solid lines represent weighted connections. An input feature vector sequence $x = (x_{tL+1}, ..., x_t)$ with sequence length L[1] is passed to a stack of multiple recurrently connected hidden layers through weighted connections to compute the hidden vector sequences $h = (h_{tL+1}, ..., h_t)$ and consequently, the output vector sequence $y = (y_{tL+1}, ..., y_t)$ which represent the predictions of future sensor readings through the learnt model.

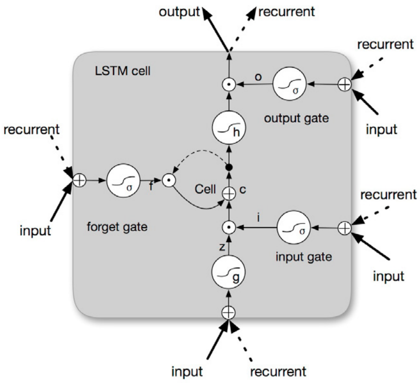

Fig. 3: Schematic of LSTM unit.

Unlike RNN, LSTM addresses the problem of long-term dependencies by additionally introducing a purpose-built memory unit to store information of previous time steps. The basic structure of LSTM memory unit is composed of three essential gates and a cell state. As shown in Figure 3, the memory cell contains the information it memorized at time t, the state of the memory cell is bound up together with three gates, the input vector of each gate is composed of the input part and the recurrent part. A forget gate controls what to abandon from the last moment, the input gate decides what new information will be stored in the cell state, while, the output gate decides which part of the cell state will be output and the recurrent part is updated by current cell state and fed into the next iteration [14].

The formulas for updating each gate and cell state are defined as follows:

$$z^t = g(W_z x^t + R_z y^{t-1} + b_z) \quad (1)$$

$$i^t = \sigma(W_i x^t + R_i y^{t-1} + p_i \odot c^{t-1} + b_t) \quad (2)$$

$$f^t = \sigma(W_f x^t + R_f y^{t-1} + p_f \odot c^{t-1} + b_f) \quad (3)$$

$$c^t = i^t \odot z^t + f^t \odot c^{t-1} \quad (4)$$

$$o^t = \sigma(W_o x^t + R_o y^{t-1} + p_o \odot c^t + b_o) \quad (5)$$

$$y^t = o^t \odot h(c^t) \quad (6)$$

[1] The sequence length L can be arbitrary long in order to memorize a very long historic sequence. However, there is a trade-off with the computational resources and the time required to train the model.

Here $x^t \in R^d$ and $y^t \in R^d$ are input and output vector of the unit at time t, $W_k(k = z, i, f, o)$ and $R_k(k = z, i, f, o)$ are weight matrices for input part and recurrent part of different gates, $b_k(k = z, i, f, o)$ denotes bias vector and the functions σ, g and h are non-linear functions such as sigmoid or tanh, $\odot$ means point-wise calculation of two vectors. For completeness, we add $p_k(k = i, o, f)$ to the formulas, which denote peephole connection and is mostly used in LSTM variants.

3.2. The Sense Module

The *Sense Module* consists of a set of functionalities that are responsible for acquiring, handling and transmitting data streams that are typically being generated from sensors installed at the industry's production line. The sensor data are stored in real-time in the framework's central InfluxDB [15] database. The InfluxDB database is open source and is used to optimize and integrate time-series data. Analytics and insights generated in real-time are also stored as InfluxDB data in the form of time-series. The integration points between the sensors and the framework have been achieved by using an underlying Kafka [19] platform. Along with the centralized storage database for sensor data, categorization filters are also available for efficient distribution of each variable to the category that belongs to. Pre-processing filters are used to handle the raw time-series data in order to then analyse them more easily. These filters include: (i) the *replacement* of missing values by their mean, so incomplete data points can still be used in further analysis; (ii) the *discretisation* of data where a numeric attribute is converted to a nominal one, treating all values in an automatically determined interval as being the same nominal data; and (iii) the *normalisation* of data which transforms a numeric attribute to a different interval. In such a way, variables are curated, harmonised and categorized under specific parts of the technical equipment, which facilitates the consistent identification of specific malfunctions in the production line. It also enables the automatic semantic alignment of the collected data. In Figure 4, we present the processing pipeline of the Sense Module.

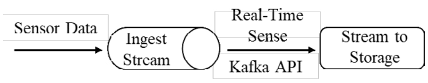

Fig. 4: The Sense Module Pipeline.

3.3. The Detect Module

The *Detect Module* consists of a sequence of "State Detection" and "Health Assessment" steps. It includes real-time statistical/machine learning algorithms embedded in an appropriate and continuously processing software in order to recognize the presence of an unusual (and potentially hazardous) state within the behaviours or activities of the monitored system, with respect to some model of 'normal' behaviour which are either learned from real-time data observations or labelled from domain experts. The detection models continuously learn from the actual equipment behaviour by updating and improving the incorporated diagnostic models by using a LSTM model. The

Detect Module is then able to recognise and categorise different states from the data. The detection models can be designed to provide more accurate predictions of certain behaviours or to provide a 'snapshot' to a specific behaviour. In Figure 5, we present the processing pipeline of the Detect Module.

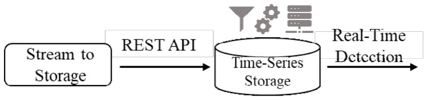

Fig. 5: The Detect Module Pipeline.

Besides, the *Detect Module* provides functionality to execute queries on the time-series data. Some of the executed queries include:

- The provision of a list of applicable sensor streams for a later analysis based on the preferred data values (e.g. "Give me all sensor streams with an acceleration sensor value more than X.");
- The provision of a subset of the data within each previously selected sensor stream, e.g. based on a time range. Standard time-series queries are applied here (e.g. "Give me for all sensor streams I previously found the raw data of the last 5 days.").

3.4. The Predict Module

The *Predict Module* includes state prediction of a whole system or components with respect to the mechanical system, i.e. prediction about the time-to-failure and the probability distribution function of the failure occurrence against to the projected performance level for each component of the system assuming no maintenance actions. The analysis is carried out by different algorithms and essentially by combination of data-driven algorithms in conjunction with physical models. Therefore, a LSTM model is used taking into account the data and information gathered in the InfluxDB along with maintenance actions performed in order to continuously update the prognostic models. The predictive models are able to provide a real-time estimate of the current performance level for the system while simultaneously forecasting the predicted performance levels based on current and time-framed historical data. In Figure 6, we present the processing and analysis pipeline of the Predict Module.

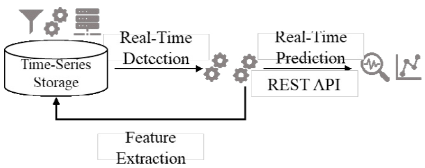

Fig. 6: The Predict Module Pipeline.

The phases of the algorithm are listed in Algorithm 1. The Deep Analysis Framework takes as input in the form of Time-Series Streaming Sensor Data S and Historical Sensor Data (H_{train}, H_{test}) and the Number of training epochs e and gives as output a set of Time-Series Predictions R in real-time. Initially, iterating over e epochs (Lines 3 - 15), the proposed LSTM model is trained over historical sensor data H_{train}, with the standard backpropagation through time (BPTT) algorithm (Lines 4-7). Afterwards (Lines 8 - 14), the *RMSE* and *MAE* error metrics are calculated on both seen and unseen data (H_{train} and H_{test} respectively) and their convergence is evaluated. If the test error stops decreasing after a certain training epoch, means that the LSTM model has reached its *generalization potential*, thus the training stops. The final trained model is then stored (Line 16) in the framework's central InfluxDB database. In the following, for a sequence of consecutive time-series (Lines 21 - 26), the algorithm filters sensor values by distributing each value in a range category that it belongs to (Line 22). Then, the algorithm applies the already trained LSTM model (i.e. binary classification for damage cf. Figure 8 or regression for RUL cf. Figure 11) in the filtered data (Line 23), stores them in the InfluxDB (Line 24) and concatenates existing with future predictions (Line 25). Finally, the algorithm returns the predictions as time-series (Line 28).

Algorithm 1: Deep Analysis Framework

Input: Real-Time Sensor Data S, Historic Sensor Data (H_{train},H_{test}), Number of training epochs e
Output: A set of Time-Series Predictions R

1 **begin**
2 /*Initially the proposed LSTM model is trained on historical sensor data*/
3 **for** ($i \leq e$) **do**
4 **foreach** ($H_k \in H\_train$) **do**
5 FORWARD_PROPAGATE $model(H_k)$
6 BACKWARD_PROPAGATE $model(H_k)$
7 **end**
8 /*Measure model's performance in each epoch*/
9 $RMSE_{train}$, MAE_{train} = $model(H_{train})$
10 $RMSE_{test}$, MAE_{test} = $model(H_{test})$
11 **if** $RMSE_{train,test}$, $MAE_{train,test}$ *converged* **then**
12 /*Stop training*/
13 **break**
14 **end**
15 **end**
16 STOREMODEL(model)
17 **end**
18 **begin**
19 /*The Deep Analysis Framework receives Real-time Sensor Data as input and the Trained Model*/
20 LOADMODEL(model)
21 **foreach** ($S_i \in S$) **do**
22 $F_k \leftarrow$ FILTER(LONGHISTORICSEQUENCE($S_{i-w}, \ldots, S_{i+w}$))
23 $R_k \leftarrow model(F_k)$
24 STORE(R_k)
25 CONCATENATE(R, R_k)
26 **end**
27 /*Time-Series Predictions are sent as output*/
28 $return(R)$
29 **end**

4. Experimental Evaluation

In this section, we describe our approach to model and predict the behavior of a real-world manufacturing system, consisting of several components with embedded sensors. Modeling is a basic and fundamental design approach to solving real-world problems. Our goal is to validate the efficiency of the proposed framework based on AI principles in order to provide better predictability of the real-world system.

4.1. Data Preliminaries

The dataset we used is coming from monitoring the core machine of the production line. In this setting also served as the preliminary design basis for the setup of the entire pipeline of the proposed architecture by installing, configuring, integrating and validating the *Sense Module*, *Detect Module* and *Predict Module*. The dataset comprises 2, 976, 665 readings from 4 sensors installed in the core machine, measuring *acceleration*, *velocity*, *overall bearing* and *shock*. We have also the *timestamp* and the *diameter* of the material that passes through the monitored machine in real-time. Finally, the dataset is enriched with the *damage* which indicates whether a damage exists or not. The damage is the only attribute which is collected once a month and aligned periodically coming from planned maintenance activities.

We developed 2 kinds of deep learning models. The first is a neural network consisted of LSTM units in the support of binary classification (i.e. damage or not) by applying a binary cross entropy loss function. The second model is an LSTM network for regression, predicting the Remaining Useful Life of the monitored machine. This means that the data are being processed differently in each case. For the binary classification model, we take into consideration the identifiers of the machines which behaviour naturally and degrade after some regular cycles (i.e. after some millions of timestamps). For the regression problem, we exploit only the identifiers of the machines that demonstrate frequent damages in order to model the Remaining Useful Life which acts as label for the regression problem. In both cases, we make predictions for multiple machines, not only for one. The data have been collected over the period of 2.5 months, with sampling rate $0.1Hz$ (i.e. one reading per 10 seconds). After the above mentioned processing and rescaling of values in the range of [0, 1], the binary classification problem consists of 234, 831 training samples and 71, 334 test samples. The regression problem consists of 196, 374 training samples and 71, 517 test samples. It should be mentioned that no shuffling is used because both problems handle time-series data, so the test set consists of the last samples of the sequences. After experimentation with all the provided features, both models performed better using the *acceleration*, the *diameter* and the label which expresses the *damage* for the binary classification model and *RUL* for the regression model.

4.2. Model Fitting

The best way for picking the most appropriate neural network architecture and the training parameters for the desired results is through the hyper-parameter optimization algorithms. After tuning of the number of LSTM units, batch size, learning rate, activation function, optimizer and dropout probabilities, we selected two LSTM architectures for the binary classification and regression problem, respectively. Both of the architectures reside in the same flowchart (cf. Figure 7) for training and testing. In the inference process, we deploy the model as a RESTful API by using the TensorFlow serving API and making requests using the test data.

4.2.1. Damage Prediction

As far as the binary classification architecture (cf. Figure 8) is concerned, we trained the model by using batch size of 512,

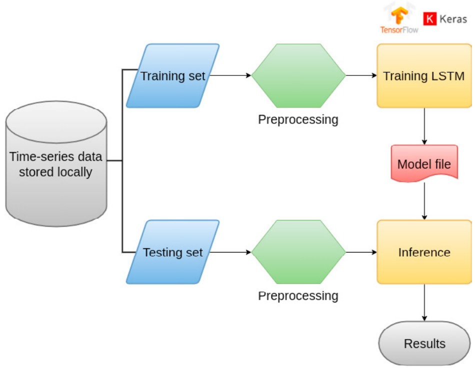

Fig. 7: Flowchart for training and testing.

learning rate of 0.001, 5 epochs, the tanh activation function and the sigmoid activation function after the dense layer. We also used the binary cross-entropy cost function, the adam optimizer, 0.5 dropout probability after each LSTM sequence and a window of 100 time-series. The input is processed in a batch mode, which refers to the mini-batch of samples in each training step, multiplied by the time-series window which is 100 time steps and finally multiplied by the number of features which are 3, namely diameter, acceleration and damage.

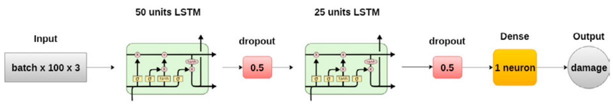

Fig. 8: LSTM architecture for damage prediction.

The results acquired refer to the Model Accuracy Diagram (cf. Figure 9) depicting the training and testing accuracy after each epoch and also the predictions over the test set (cf. Figure 10) compared with the ground-truth values. These figures demonstrate that the test set converges with the training set within 1 epoch and that the damage prediction is computed with high accuracy over the ground-truth data. As we are handling time-series data for this binary classification task, no shuffling is used and therefore the model converges within very few epochs. Upon this convergence, we limit up to 5 more training cycles the model fitting in order to omit overfitting and achieve at the same time model generalisation.

We also computed the confusion matrix (Table 1), extracting precision, recall and accuracy metrics (Table 2).

Table 1: Confusion matrix of damage prediction.

	Normal	*Damage*
Normal	45804	25
Damage	26	25479

Table 2: Metrics for damage prediction.

Precision	*Recall*	*Accuracy*
99.9%	99.89%	99.92%

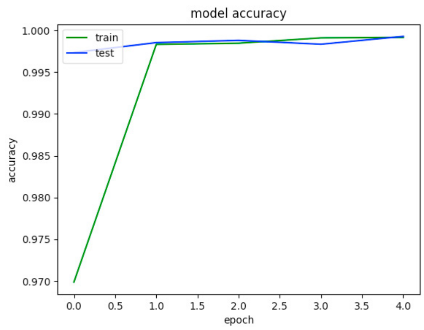

Fig. 9: Training-testing accuracy after each epoch.

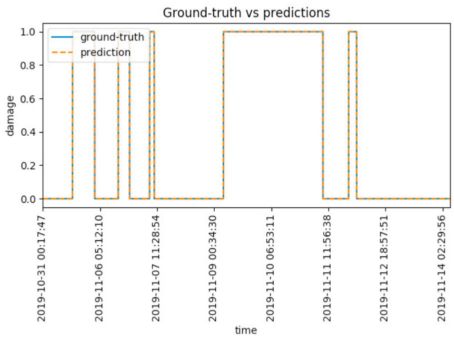

Fig. 10: Predictions vs ground-truth values on test set.

4.2.2. *Remaining Useful Life Prediction*

As far as the regression architecture is concerned (cf. Figure 11), we trained the model by using batch size of 512, learning rate of 0.001 decreased every 4 epochs by a factor of 0.4, 30 epochs, the rectified linear unit activation function, the mean squared error cost function, the adam optimizer, 0.5 dropout probability after each LSTM sequence and a window of 100 time-series.

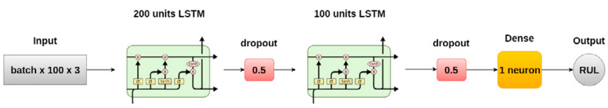

Fig. 11: LSTM architecture for RUL prediction.

The results acquired refer to the MSE (Mean Squared Error) Diagram (cf. Figure 12) depicting the training and test MSE after each epoch and also the predictions on the test set (cf. Figure 13) compared with the ground-truth values. These figures demonstrate that the model well converges within 25 epochs and RUL prediction is computed with high accuracy over the ground-truth data. In this regression task, it requires more epochs to converge over the time-series data. No shuffling is used because we wanted to predict RUL through a robust model which learns well over the data as the time-series or events evolve. Also, potential degradation and variations measured by the sensors at the monitored machine contribute in RUL prediction across the time. Upon convergence, we limit up to 5 more training cycles the model fitting in order to omit overfitting and achieve at the same time model generalisation.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

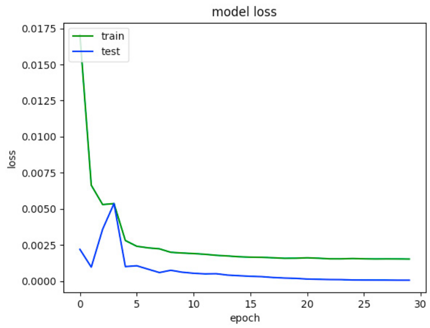

Fig. 12: Training-Test MSE after each epoch.

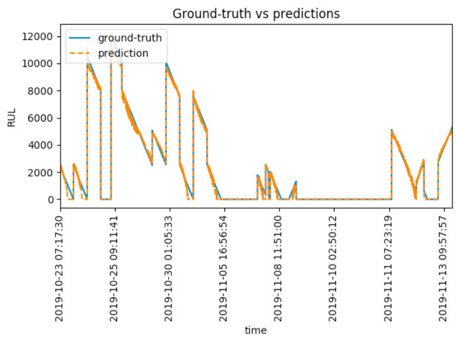

Fig. 13: Predictions vs ground-truth values on test set.

5. Conclusions and Future Work

The purpose of this work is to present a systematic analysis in the direction of Predictive Maintenance assisted by real-time monitoring and unexpected events prediction. Deep learning, as a sub-field of Machine Learning and Artificial Intelligence, is serving as a bridge between big machinery data and data-driven machine health monitoring. Therefore, within the past years, they have been applied in various machine health assessment tasks. The proposed LSTM-based Deep Learning architecture gives very promising results in predicting the sensor behavior and unexpected events such as damages and assets RUL.

In the near future, we plan to extend our data-driven solution for predicting the type of failure given the modeled behavior of an industrial system by exploiting the available historical maintenance data (e.g. kind of failures of working parts in machinery, regular maintenance activities to equipment, labelled time frames with semantically enriched type of failures, etc.). To this end, we will frame the task of failure prediction as a sequential multi-class classification problem by also experimenting over larger data sets with more records. Towards this direction, we will augment our approach for predicting concrete failure events within an adequately-large time window, thus allowing for effective, purely data-driven root cause analysis along with a set of recommendations for maintenance actions before a damage occurs.

Acknowledgements

The research leading to these results has received funding by the European Commission project H2020 UPTIME "Unified Predictive Maintenance System" (https://www.uptime-h2020.eu) under grant agreement No. 768634.

References

[1] Adhikari, R., 2015. A neural network based linear ensemble framework for time series forecasting. Neurocomputing 157, 231–242.
[2] Bandaragoda, T.R., Ting, K.M., Albrecht, D., Liu, F.T., Wells, J.R., 2014. Efficient anomaly detection by isolation using nearest neighbour ensemble, in: 2014 IEEE International Conference on Data Mining Workshop, IEEE. pp. 698–705.
[3] Charette, R., Romero, J., 2015. The staggering impact of it systems gone wrong. IEEE Spectrum .
[4] Cheng, Y., Zhu, H., Wu, J., Shao, X., 2019. Machine health monitoring using adaptive kernel spectral clustering and deep long short-term memory recurrent neural networks. IEEE Transactions on Industrial Informatics 15, 987–997.
[5] Chi, Q., Yan, H., Zhang, C., Pang, Z., Xu, L.D., 2014. A reconfigurable smart sensor interface for industrial wsn in iot environment. IEEE Transactions on Industrial Informatics 10, 1417–1425. doi:10.1109/TII.2014.2306798.
[6] Ellefsen, A.L., Bjørlykhaug, E., Æsøy, V., Ushakov, S., Zhang, H., 2019. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. Reliability Engineering & System Safety 183, 240–251.
[7] Elsheikh, A., Yacout, S., Ouali, M.S., 2019. Bidirectional handshaking lstm for remaining useful life prediction. Neurocomputing 323, 148–156.
[8] Giurgiu, I., Schumann, A., 2019. Explainable failure predictions with rnn classifiers based on time series data. ArXiv abs/1901.08554.
[9] Goh, S.T., Rudin, C., 2014. Box drawings for learning with imbalanced data, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM. pp. 333–342.
[10] Graves, A., Schmidhuber, J., 2009. Offline handwriting recognition with multidimensional recurrent neural networks, in: Advances in neural information processing systems, pp. 545–552.
[11] Guillén, A.J., Crespo, A., Gómez, J.F., Sanz, M.D., 2016a. A framework for effective management of condition based maintenance programs in the context of industrial development of e-maintenance strategies. Computers in Industry 82, 170–185.
[12] Guillén, A.J., Crespo, A., Macchi, M., Gómez, J., 2016b. On the role of prognostics and health management in advanced maintenance systems. Production Planning & Control 27, 991–1004.
[13] Guo, Y., Wu, Z., Ji, Y., 2017. A hybrid deep representation learning model for time series classification and prediction, in: 2017 3rd International Conference on Big Data Computing and Communications (BIGCOM), IEEE. pp. 226–231.
[14] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9, 1735–1780.
[15] InfluxDB, 2019. Available online: https://www.influxdata.com/ .
[16] Jansen, F., Holenderski, M., Ozcelebi, T., Dam, P., Tijsma, B., 2018. Predicting machine failures from industrial time series data, in: 2018 5th International Conference on Control, Decision and Information Technologies (CoDIT), IEEE. pp. 1091–1096.
[17] Jeschke, S., Brecher, C.P.D., Song, H., Rawat, D.B., 2017. Industrial internet of things and cyber manufacturing systems.
[18] Jung, D., Zhang, Z., Winslett, M., 2017. Vibration analysis for iot enabled predictive maintenance, in: 2017 IEEE 33rd International Conference on Data Engineering (ICDE), IEEE. pp. 1271–1282.
[19] Kafka, A., 2019. Available online: https://kafka.apache.org/ .
[20] Liao, L., Ahn, H.i., 2016. Combining deep learning and survival analysis for asset health management. International Journal of Prognostics and Health Management .
[21] Liu, F.T., Ting, K.M., Zhou, Z.H., 2012. Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD) 6, 3.
[22] Metzger, A., Föcker, F., 2017. Predictive business process monitoring considering reliability estimates, in: International Conference on Advanced Information Systems Engineering, Springer. pp. 445–460.
[23] Prić, D., Nedic, N., Stojanovic, V., 2017. A nature inspired optimal control of pneumatic-driven parallel robot platform.
[24] Reddy, K.K., Sarkar, S., Venugopalan, V., Giering, M., 2016. Anomaly detection and fault disambiguation in large flight data: a multi-modal deep auto-encoder approach, in: Annual Conference of the Prognostics and Health Management Society.
[25] Sikorska, J., Hodkiewicz, M., Ma, L., 2011. Prognostic modelling options for remaining useful life estimation by industry. Mechanical systems and signal processing 25, 1803–1836.
[26] Susto, G.A., Schirru, A., Pampuri, S., McLoone, S., Beghi, A., 2015. Machine learning for predictive maintenance: A multiple classifier approach. IEEE Transactions on Industrial Informatics 11, 812–820.
[27] Wang, F., Liu, J., 2011. Networked wireless sensor data collection: Issues, challenges, and approaches. IEEE Communications Surveys Tutorials 13, 673–687. doi:10.1109/SURV.2011.060710.00066.
[28] Wang, F., Liu, X., Deng, G., Yu, X., Li, H., Han, Q., 2019a. Remaining life prediction method for rolling bearing based on the long short-term memory network. Neural Processing Letters , 1–18.
[29] Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L., 2019b. Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters 119, 3–11.
[30] Wang, Y., Chen, Q., Hong, T., Kang, C., 2019c. Review of smart meter data analytics: Applications, methodologies, and challenges. IEEE Transactions on Smart Grid 10, 3125–3148.
[31] Yu, M., Wang, D., Luo, M., 2014. Model-based prognosis for hybrid systems with mode-dependent degradation behaviors. IEEE Transactions on Industrial Electronics 61, 546–554.
[32] Zaarour, T., Pavlopoulou, N., Hasan, S., Curry, E., et al., 2017. Automatic anomaly detection over sliding windows: Grand challenge, in: Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems, ACM. pp. 310–314.
[33] Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P., Gao, R.X., 2019. Deep learning and its applications to machine health monitoring.