A Comparison of Machine Learning Algorithms and Tools in Prognostic Predictive Maintenance: a Focus on Siamese Neural Network Models*

Lazzarinetti Giorgio[1 [0000−0003−0326−8742]], Massarenti Nicola[1 [0000−0002−8882−4252]], Mantisi Stefania[1 [0000−0003−4446−9743]], and Sasso Onofrio[1 [0000−0003−3288−777X]]

Noovle S.p.A, Milan, Italy https://www.noovle.com/en/

Abstract. With the advent of Industry 4.0, predictive maintenance techniques have largely spread throughout companies. However, it is still difficult to understand how to implement a predictive maintenance strategy to get satisfactory results. In this research we propose a methodology to define a benchmark in terms of performance of machine learning algorithms in the context of prognostic predictive maintenance from a classification perspective. In defining such a benchmark we use three target datasets publicly available over which to compare different preprocessing and feature engineering techniques and different machine learning algorithms and auto learning tools. Our benchmark shows that it is possible, by following the guidelines delineated in this paper, to select the proper combination of preprocessing, feature engineering and algorithms/tools to get an average F1-score of 98%. Moreover, we propose an innovative approach based on siamese neural networks that shows comparable results with respect to the benchmark defined, thus showing that also this kind of algorithm has to be tested to be sure to reach the best possible results.

Keywords: Predictive Maintenance · Benchmark Definition · Siamese Neural Network.

1 Overview

Thanks to the advent of Industry 4.0 and the enhancements in machine learning techniques, in recent years predictive maintenance (PdM) applications have largely spread throughout companies. Since PdM is an active area of research,

* Activities were partially funded by Italian "Ministero dello Sviluppo Economico", Fondo per la Crescita Sostenibile, Bando "Agenda Digitale", D.M. Oct. 15th, 2014 - Project n. F/020012/02/X27 - "Smart District 4.0".

there are thousands of papers published on the topic, however among all the possibilities of implementing a PdM system, it is still difficult to identify a specific strategy to get satisfactory results. On the shake of this, this research aims at defining a technical benchmark to measure the performance of different fault prognosis algorithms and auto-learning tools in the context of prognostic PdM. More precisely, this research is driven by the business need of a partner company that produces vertical cutting machines and aims at creating a PdM system to predict breakage events, thus reducing related costs by avoiding them. One of the main issues of the partner company is that they have scarse connectivity throughout the production line, so they cannot stream data in real-time to the final system that will be cloud-based. In this context, the final goal is to produce a system capable of monitoring in semi-real-time through sensors some operating parameters of the machines in order to be able to predict their remaining useful life (RUL). Generally, the RUL is a continuous variable that requires a regression problem to predict it. However, the semi-real-time scenario poses the issue of interacting with the machine at discrete time intervals sending a batch of the collected data every hour. Thus, since in the real case scenario we will get data in batch with a certain time delay, in this research we focus on the context of PdM from the classification point of view, i.e. we aim at predicting the class of breakage of the last observations collected. Thus, we designed a methodology to compute a benchmark for prognostic PdM based on the main state-of-the-art machine learning algorithms and the available auto learning tools to provide the partner company with a precise methodology to select the best algorithm and determine how it performs with respect to the benchmark. Moreover we compare the results with an innovative approach based on Siamese Neural Network (SNN), which is an algorithm never used in this context. Computing this benchmark will allow the partner company to define the best PdM strategy by following the guidelines defined in this paper and to check whether the results they are obtaining by implementing the PdM system are competitive with the state-of-the-art results. The rest of this paper is organized as follows. In chapter 2 a comprehensive analysis of the state of the art for PdM is presented. Then in chapter 3 the methodology for the creation of the benchmark is defined together with the innovative approach based on SNN. In this chapter also the dataset used for the computation are described. In chapter 4 the results obtained are shown with a focus on the comparison of the benchmark with the innovative approach and, finally, in chapter 5 some conclusive remarks.

2 State of the Art

Predictive maintenance (PdM) [1], which is the analysis of industrial machinery's operating parameters in order to predict breakage events [2], is an active area of research that has found the right space for application only recently thanks to the fact that industry is currently moving towards the so called industry 4.0 [3]. Indeed, new technologies from industry 4.0, such as the Internet of things (IoT) [7], cloud computing and sensors and advances in artificial intel-

ligence from software [5] and hardware [8] perspectives, allow the integration of people, machines and products, thus making possible a fast exchange of information and the generation of even more data [4], enabling efficient and effective PdM for companies [6]. There are several approaches to PdM. In particular, the taxonomy of the approaches differs in three macro aspects: the architecture of the system (OSA CBM, supported by the cloud or based on industry 4.0 technologies), the objective (minimization of costs, maximization of reliability and availability or multi-objective) and the type of algorithms used [1]. In this research we focus on fault prognosis and diagnosis algorithms, in the context of industry 4.0 with the goal of maximizing the reliability and availability of the system, which can be carried out with two macro typologies of approaches: knowledge based and data driven. Knowledge based methods make use of expert knowledge of the monitored systems and can be divided into 3 categories: ontology based, rule based and model based. Ontology based approaches allow for the creation of knowledge bases for different systems and machinery [9]. In the context of PdM, several ontologies have been built [10–13], however these approaches must be used together with other reasoning techniques to be effective. Rule based approaches are based on the evaluation of data with a set of fixed IF-THEN-ELSE rules determined by domain experts, which has the advantage of including a-priori knowledge in the system [14–17]. These approaches are very effective, but clearly not scalable. Model based approaches are based on the implementation of mathematical models of the physical processes which have an impact on the health of the system components [18–22]. These approaches are applicable only when the underlying physical process can be perfectly described by a mathematical model without adopting too stringent assumptions or constraints and this rarely happens [23–25]. Data driven approaches, on the other end, are approaches that use historical data to learn a model of the system's behavior with machine learning techniques [26]. From the literature perspective, the task of predicting failures can be reduced to three main types of problems: binary classification, multi-class classification and regression. Binary classification is used in PdM with the goal of estimating the probability that after a certain number of machine cycles the machine will break down. Similarly for multi-class classification, where each class represents the probability that a machine will break down in the following N cycles, with N possibly different for each class. Regression models, on the other hand, are used to estimate the number of life cycles remaining (RUL). Some traditional machine learning algorithms used in this field are: Artificial Neural Network (ANN) [27–30], Decision Tree (DT) and Random Forest (RF) [31–33], Support Vector Machine (SVM) [34, 35] and K-nearest Neighbor (KNN) [36]. However, in recent times, research has moved towards Deep Learning algorithms that have proven to somehow outperform many of these more traditional models [1]. Specifically, the main algorithms used in the context of PdM are: Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Generative Adversarial Network (GAN). In this context, CNNs have shown enormous ability to extract useful and robust features to perform fault diagnosis [37–39]. Experiments have shown that, with

proper hyperparameters tuning, models can achieve 99% accuracy levels. CNNs are also often used to predict the RUL, as in [40, 41]. Also RNNs have often been used as a fault diagnosis tool in recent times as their consolidated ability to model time sequences has guaranteed these algorithms superior performance compared to other types of networks [42–44]. Moreover, thanks to the ability of Long Short Term Memory (LSTM) and Gate Recurrent Unit (GRU) cells to handle long and time-dependent time series, many studies have been carried out on the prediction of the RUL via these networks [45–47]. In the context of PdM also GANs have been proposed to identify the type of machine failure [48, 49] or to predict the RUL [50] by modeling the trend of the health indicators of a machinery. The advantage of these networks is that the health indicator trend model becomes more accurate as more data is collected. In recent times, also Siamese Neural Networks (SNNs) have emerged among deep learning algorithms [51]. SNNs are models composed of two parallel identical sub-networks, which process two differents input data in order to create an embedding to be compared. The two sub-networks are trained to produce an embedding so that the similarity measure between the embedding produced is minimized in case the two embedded inputs are of different classes and maximized otherwise [52]. This type of learning is known as one-shot-learning. These approaches have proven to be extremely useful also in the context of time series analysis. An example is found in [53], where a system is proposed to measure the similarity between time series using SNN with two twin subnets consisting of RNNs. The application of these algorithms in the field of PdM is still little explored today [54].

3 Benchmark definition

The goal of this research is to define a benchmark in terms of performance of the data driven PdM algorithms for fault prediction and compare the results with an innovative approach based on SNNs. As explained in Chapter 1, given that in the partner company's scenario data are send to the PdM model in batch with a certain delay, we focus on the case of multi-class classification, where the aim is that of determining, given a time series of variables, the breakage class. For the definition of the benchmark, not only custom algorithms are used, but also some auto-learning tools such as Google Cloud AutoML [55] and Google Cloud BigQuery ML [56] (BQ-ML). In order to define this benchmark we use three widely used public datasets and calculate the performance for each of them, comparing different preprocessing and feature engineering approaches. In the following we provide all the details of each step necessary to compute the benchmark. We also define also the implementation and evaluation of the algorithm based on SNN and the calculation of the positioning of the SNN based approach with respect to the benchmark on public datasets.

3.1 Datasets description

Three different public datasets have been identified for the definition of the benchmark.

Zenodo predictive maintenance dataset The dataset published by Zenodo [57] consists of a series of IoT sensors for predictive maintenance in the elevator industry. The collected data can be used for predictive maintenance of elevator doors in order to reduce unscheduled stops and optimize maintenance interventions.The dataset contains operational data in the form of time series sampled at a frequency of 4Hz. In particular, for each lift there are electromechanical sensors, physical sensors and environmental sensors. In the following we will refer to this dataset as OML.

NASA Turbofan Engine Degradation dataset The dataset published by NASA's Prognostic CoE [58] concerns the degradation of aeration engines and is constructed using C-MPASS. The dataset is simulated under different combinations of operating conditions and for different types of faults and contains several variables that describe the characteristics of the evolution of the fault. In the following we will refer to this dataset as Aircraft.

XJTU-SY Bearing Datasets The XJTU-SY dataset [59] was published by the Institute of Design Science and Basic Components at Xi'an Jiaotong University, China for the predictive maintenance of rotating elements. The dataset contains multivariate time series of 15 rolling bearings from start to failure, acquired by conducting several accelerated degradation experiments. In the following we will refer to this dataset as XJTU-SY.

3.2 Preprocessing and Feature engineering

In order to define a precise benchmark, we decided not only to compare different machine learning algorithms but also different preprocessing techniques and different feature engineering approaches. In this case, three distinct feature engineering modes have been defined that share a common preprocessing phase.

Preprocessing Firstly data is normalized and the features that have zero variance are eliminated. The target variable is then defined as RUL, i.e. number of cycles missing from the fault. Since the benchmark to be determined refers to a multi-class classification problem we need to convert the RUL from a continuous variable to a discrete variable. Thus, we define a methodology to divide the dataset into 3 classes: one containing the cases of RUL between 0 and N, one containing the cases of RUL between N and M with M>N and one with the cases of RUL greater than M. In this way the three classes represent a case of failure in the short term (RUL<N), a case of failure in the medium term (N<RUL<M) and a case of good functioning (RUL>M). The method of selecting the parameters N and M depends both on the use case, i.e. on how long the cycles last and in how many cycles on average a failure case occurs, and on the performance of the models, i.e. on the accuracy of the models in the short, medium and long term. Starting from a given N (the minimum number of cycles that ensure the

operator room to maneuver), a series of machine learning models is trained for larger values of N. For each model trained, the performance are calculated and the performance trend is studied as N varies. The idea is to select N as large as possible (to ensure that the prediction occurs in time, guaranteeing the operator room to maneuver) but with the aim of maintaining good performance. A limit deviation of 5% from the maximum performance value is therefore considered. Once parameter N has been selected, M is selected in the same way.

Feature Cycle (FC) The first feature engineering approach consists in using the preprocessed dataset and in creating for each feature and for each pair of features the corresponding second degree feature crosses (for example, given the features x and y, a dataset containing x, y is considered, xy, x^2y^2). This is a standard feature engineering technique that has to be tested in order to find the best solution possible. Once the features are generated, since some datasets contain many sensors and the size of the features may explode, the features are reduced through Principal Component Analysis (PCA) , in order to represent the features through an embedding vector. The vector is constructed by taking only the k principal components that describe at least 95% of the variance present in the data.

Feature Rolling (FR) A second method of feature engineering consists in using the preprocessed dataset enriched with the calculation of the second degree feature crosses as described above, then adding a level of temporal aggregation to also take into account the overall trend of the series. In this case, for each detection x_i the sequence of the t values preceding x_i, $(x_{i-t}, ..., x_i)$ is taken and it is replaced with the average of the values of the sequence of length t. This allows to engineer historical information and include them in the features, letting some algorithms (which otherwise would not be able to consider historical information) take these historical correlation into account. To determine the optimal value of t, an approach based on the analysis of the performance of the models generated by considering different values of t is adopted. In this case we start from a minimum value of $t = 2$ and proceed by increasing t by 1. For each increment, a classifier is trained and the performance are measured. The choice of t must be made taking into consideration the peak point of the performance trend, but always considering that too large t implies the need to have a certain number of measurements before being able to make the prediction, therefore we take into account the average life cycle of a machine and select t as the minimum value between the peak point and $\frac{1}{3}$ of the maximum number of life cycles. Also in this case follows the reduction of the dimensionality based on PCA and on the principle of 95% of the variance explained.

Feature Rolling Enriched (FRE) The third and last approach of feature engineering starts from the pre-processed database enriched with second degree feature crosses over which FR is performed as described above but, for each batch

the features are enriched by calculating statistical indices as the mean value, the median, the minimum, the maximum, the skew, the standard deviation and the kurtosis index. This allows to enrich the dataset with statistically significant features that could help in modeling particular relations between variables. Also in this case follows the reduction of the dimensionality based on PCA and on the principle of 95% of the variance explained.

3.3 Algorithms and auto-learning tools

Once the datasets have been prepared, different machine learning algorithms are trained for each of the datasets described above which, according to the state of the art, are widely used. Specifically, the algorithms chosen for the creation of the benchmark are: Logistic Regression (LR) as baseline algorithm, RF, ANN and KNN since they are the main used algorithms in this context according to the state of the art (from which we excluded SVM due to their scalability issues with large datasets) and LSTM classifier which, among the DL techniques, is the most consolidated in sequence learning problems [1]. The aforementioned models are trained using Google Cloud Vertex AI [60], in order to keep track of models and versions and taking advantage of the hyper parameter optimization provided by Google. This optimizer called hypertune is based on Google Vizier and is a black box optimization service released in 2017 [61], based on Bayesan optimization. In addition to training the models described above, we train other models using the auto-learning tools provided by Google. Specifically: Google Cloud AutoML Tables, Google Cloud BQ-ML LR. For these tools of the Google suite it is not necessary to perform tuning of the hyperparameters because these are carried out automatically. For both the algorithms and the auto-learning tools the F1 score is calculated. The training of the algorithms is conducted in a systematic way. Each algorithm or tool is trained using the three different types of datasets.

3.4 Benchmark computation

Once all the algorithms and tools have been trained, the benchmark is computed by averaging the maximum performance obtained in terms of F1 score for each algorithm and for each feature engineering technique applied. More formally, given a set of datasets $D : \{d_1, ..., d_D\}$, algorithms $A : \{a_1, ..., a_A\}$ and feature engineering techniques $F : \{f_1, ..., f_F\}$ and considered $F1^d_{a,f}$ as the F1-score associated to the a^{th} algorithm, the f^{th} feature engineering and the d^{th} dataset, the benchmark is computed as

$$\frac{\sum_{d=1}^{D} \underset{\{a \in A, f \in F\}}{max} F1^d_{a,f}}{|D|} \tag{1}$$

Thus, this benchmark represents an average result which can be pursued by applying the correct feature engineering technique and the correct algorithm to a specific dataset.

Guidelines for benchmark computation and evaluation To summarize, given a set of dataset D, to compute the benchmark the following steps need to be performed for each dataset:

- Define the minimum number of cycles N to let the operator intervene.
- Preprocess the data and define the optimal value of N and M with the methodology defined in paragraph 3.2-Preprocessing.
- Over the preprocessed data run the feature engineering technique defined as FC, FR and FRE, defining the optimal parameter t for the FR feature engineering technique with the methodology described in 3.2-Feature Rolling (FC).
- Divide the data in train and test with an 80/20 split and train the LR, RF, ANN and KNN algorithms with the training data using Google Cloud vertex AI leveraging the HyperTune algorithm to run cross validation and hyperparameters tuning and Google auto-learning tools that automatically perform all the optimizations.
- Test each algorithm trained with the test data and measure the F1-score.
- Compute the benchmark as described in paragraph 3.4 by considering the best F1-score for each algorithm trained and feature engineering technique applied.

The benchmark thus obtained represents an average affordable result. Moreover, by following the same steps over a real dataset, it is possible also to identify the best algorithm for the specific case and compare the performance over that dataset with respect to a predefined benchmark.

3.5 SNN based algorithm

In addition to the state-of-the-art algorithms we decided to study the performance of an innovative approach based on SNN [51]. In particular, we design an SNN composed of two twin layers of LSTM neural networks. The objective of this network is to take as input distinct time series associated with machine operation that correspond to series that do not end in a fault and series that end in a fault, sampled at different distances from the fault event itself. Each series will have a class set according to the parameters N and M selected as described in section 3.2. The assignment criterion is determined by the distance of the last value of the series from the failure event in terms of cycles. On the basis of this input, the network is trained by considering distinct pairs of time series, some of the same class and some of different classes. The LSTM-based embedding layers allow to build an embedding of these series taking into account temporal dependencies. The network is trained to understand how to create these embedding so that the selected distance metric considers more similar series of the same class and more dissimilar series of different classes. The model is trained using Google Cloud Vertex AI and leveraging hypertune for learning rate, epochs and batch size. In order to make predictions, this kind of model, for each input series, must build the embedding and compare it with the embeddings of a certain number of

previously collected series for which the class is known. The prediction class for the input series is selected based on the majority voting of the classes assigned by distance from each previously collected series. It is evident, however, that the choice of the comparison series has an impact on the performance of the classifier. Thus, we design a methodology to select the best comparison series. Given a dataset of K classes $C : \{C_1, C_2, ..., C_K\}$ and N instances $X_1, ..., X_N$ such that each instance is a tensor of F features and T subsequent time instants (namely a multivariate time series) so that each component of the instance X_n is represented by $x^f_{n,t}$ with $t \in \{1, \dots, T\}$ and $f \in \{1, \dots, F\}$

$$X_n = \begin{pmatrix} x^1_{n,1} & \cdots & x^1_{n,T} \\ x^2_{n,1} & \cdots & x^2_{n,T} \\ \vdots & \ddots & \vdots \\ x^F_{n,1} & \cdots & x^F_{n,T} \end{pmatrix}, \forall n \in \{1, \dots, N\} \tag{2}$$

For each feature f and for each time instant t, the centroid $\overline{\mu_k}$ of the class k with the components $\mu^f_{k,t}$ are computed as

$$\overline{\mu_k} = \begin{pmatrix} \mu^1_1 & \cdots & \mu^1_T \\ \vdots & \ddots & \vdots \\ \mu^F_1 & \cdots & \mu^F_T \end{pmatrix}, \mu^f_{k,t} = \frac{1}{|C_k|} \sum_{X_j \in C_k} x^f_{j,t}, \forall k \in \{1, \dots, K\} \tag{3}$$

Starting from the centroids of each class, we propose to use an approach based on KNN to get the S multivariate time series closest to each centroid and use these as comparison samples to detect the final class by majority voting. In this way we assure to use as a comparison sample just the series that are more descriptive of each class. The size of S will be determined on the basis of performance, starting from a minimum value of 1, which correspond to an n-way one shot learning, and increasing incrementally. The drivers for the choice of S will be both the performance of the models and the prediction times: since the system must be industrialized it will be necessary that these times remain relatively low, approximately in the order of seconds. Furthermore, since this is a classification model, the same metrics used to evaluate the other models that contribute to the definition of the benchmark are used to evaluate and compare the performance of this innovative approach with the benchmark.

4 Experimental results

In the following the experimental results obtained are shown. In particular, firstly the results obtained in running the methodology for N, M and t selection, as explained in Paragraph 3.2. Then, the actual results of the different models and tools trained to compute a benchmark and a comparison between the benchmark and the SNN algorithm are presented.

4.1 Preprocessing and Feature engineering

The first step for the definition of the benchmark is that related to the preprocessing and feature engineering step. It is, indeed, important to define the parameters N, M and t as described in section 3.2-Preprocessing, in order to get the best results by keeping the model useful from a business perspective (i.e., the parameters N and M do not have to be too small, otherwise the operator does not have time to stop the machine and avoid breakage event and, similarly, the t parameter does not have to be too large, otherwise it is necessary to have a lot of values in the past to be able to perform a prediction). In order to determine the parameters N and M we train several RF classifiers firstly for subsequent values of N and then, once N has been defined, of M, keeping N fixed. In Figure 1 the results for different values of N and M for the Aircraft dataset are shown. Clearly, the selection of N and M varies according to the dataset used. As an example, for the XJTU-SY dataset and the OML dataset, the breakage events happen after thousands of cycles, thus N and M are of the order of 50 to 100 thousands, while for the Aircraft dataset the breakage events happen after hundreds of cycles, thus N and M are of the order of 10 to 100 cycles. After looking at the results, and considering a limit deviation of 5% from the maximum performance value to keep N and M as large as possible, the selected parameters are defined in Table 1.

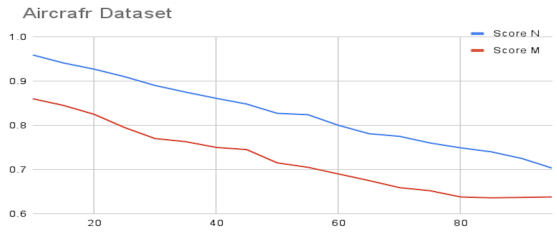

Fig. 1. N-M selection

Table 1. Best N, M and t for each dataset

Dataset	Best N	Best M	Best t
Aircraft	25	20	20
XJTU-SY	10000	10000	80
OML	100000	80000	3

Analogous considerations hold for the selection of t. A RF classifier has been trained for subsequent values of t, starting from t=2. Also in this case, the frequency of the breakage events impact the selection of t. In Figure 2 it is

shown the trend of the scores register for the Aircraft dataset varying t. In Table 1 we can see the best selected t considering a limit deviation of 5% from the maximum performance value to keep t as small as possible. The selected parameters are those used to train the different models and tools.

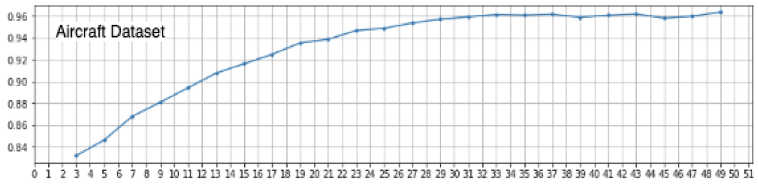

Fig. 2. t selection

4.2 General results: benchmark definition

To train and test each model and tools we applied an 80-20 split of training and test data, performing a stratified sampling. All the models have been trained on Google Cloud Vertex AI, exploiting the hyperparameters optimization module. As far as the auto learning tools are concerned, they automatically perform the hyperparameters optimization. In order to compare the results and evaluate the performance of each algorithm, we used the F1 score. In Table 2 a comparison of the results obtained is shown. By analyzing the results we can see that generally the performance over the OML dataset are better than the other and the ones over the XJTU-SY dataset are worse. On average, the RF algorithm reaches the best performance over all datasets (with all the feature engineering techniques applied), with an 85.5% of F1 score. Moreover, we can state that the FRE feature engineering technique, in most cases extremely improves the results of the algorithms. This can be seen especially in the case of Aircraft and XJTU-SY, where the performance over the other feature engineering techniques are definitely worse, but also in the case of OML, even though the performance are good also in the other cases. Finally, by considering the best results for each dataset over all the feature engineering techniques and over all the algorithms tested, on average we can state that it is possible to reach a 98% of F1-score, with a 1.7% of standard deviation. This is the benchmark computed as in Equation 1, which describes the target result that one can achieves by properly selecting the algorithm and the feature engineering technique for his own specific dataset.

4.3 SNN results

In order to define the performance of the SNN algorithm, firstly we need to define the optimal number S of comparison series. To define the optimal number S, we follow the methodology proposed in Paragraph 3.5. We compute different F1

Table 2. Final results in terms of F1-score

Dataset	Feat. Eng.	ANN	KNN	LSTM	LR	RF	AutoML	BQ-ML	SNN
	FC	0.6	0.58	0.6	0.6	0.59	0.6	0.56	0.66
Aircraft	**FR**	0.6	0.93	0.56	0.59	0.91	0.74	0.51	0.66
	FRE	0.89	0.9	0.59	0.57	0.96	0.89	0.53	0.99
	FC	0.48	0.46	0.46	0.4	0.45	0.49	0.33	0.4
XJTU-SY	**FR**	0.26	0.84	0.54	0.48	0.83	0.69	0.44	0.53
	FRE	0.86	0.87	0.93	0.57	0.99	0.42	0.31	0.89
	FC	0.97	0.99	0.97	0.88	0.99	0.99	0.81	0.8
OML	**FR**	0.91	0.99	0.96	0.9	0.99	0.99	0.95	0.82
	FRE	0.99	0.99	0.99	0.98	0.99	0.99	0.83	0.98

score based on a varying number of comparison series from 1 to 15. We compare the results over each feature engineering and select the smallest S corresponding to the best result. The best results obtained are $S = 4$ for FC, $S = 5$ for FR and $S = 5$ for FRE. Given the benchmark defined previously, we can compare the results of the SNN based algorithms. In Table 2 the results, in terms of F1 score are shown. To compute the predictions, we get the top S series closest to the centroid, defined as in Equation 3. To get the top S series we use the approach based on KNN described in Paragraph 3.5. As we can see, also in this case, the best results are obtained with the FRE feature engineering technique. Very good results are obtained for the Aircraft and the OML dataset, while acceptable results are obtained for the XJTU-SY dataset. Computing the average of the best results also for this algorithm, we have an average F1-score of 95.3%. This is slightly under the benchmark previously defined, however results are comparable, meaning that also this kind of algorithm can be adopted in the context of PdM.

5 Conclusion and future works

In this research we present a methodology for the definition of a benchmark in terms of performance in the context of prognostic predictive maintenance from the classification perspective. In defining the benchmark we consider different preprocessing and feature engineering techniques and different machine learning algorithms and auto-learning tools and we compute the benchmark over different public datasets. We also define some approaches to automate parameters selection that contribute to reach the best performance. In conclusion, we show that, despite the input dataset, it is possible to select the proper feature engineering technique and the proper machine learning algorithm or tool to reach an average F1-score of 98%. Moreover we test an innovative approach based on SNN and we show that it is competitive with the benchmark computed. To enhance the research, it could be interesting to expand the definition of the benchmark also to real datasets, to understand whether the results obtained with public dataset (some of which are synthetic) can be compared with the results obtained with real dataset.

References

1. Yongyi, R., Xiaoxia, Z., Pengfeng, L., Yonggang, W., Ruilong, D.: A Survey of Predictive Maintenance: Systems, Purposes and Approaches. ArXiv preprint, arXiv:1912.07383 (2019)
2. Paolanti, M., Romeo, L., Felicetti, A., Mancini, A., Frontoni, E., Loncarski, J.: Machine Learning approach for Predictive Maintenance in Industry 4.0. In: Proceedings of the 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA). IEEE, Oulu, Finland (2018)
3. Borgi, T., Hidri, A., Neef, B., Naceur, M. S.: Data analytics for predictive maintenance of industrial robots. In: International conference on advanced systems and electric technologies (IC_ASET), pp. 412–417 (2017)
4. Rauch, E., Linder, C., Dallasega, P.: Anthropocentric Perspective of Production before and within Industry 4.0. Computers & Industrial Engineering. **139** (2019)
5. Carvalho, T., Soares, F., Vita, R., Francisco, R., Basto, J.: A systematic literature review of machine learning methods applied to predictive maintenance. Computers & Industrial Engineering **137** (2019)
6. Nguyen, K. A., Do, P., Grall, A.: Multi-level predictive maintenance for multi-component systems. Reliability Engineering System Safety **144**, 83–94 (2015)
7. Habib, U. R. M., Ahmed, E., Yaqoob, I., Hashem, I., Ahmad, S., Imran, M.: Big Data Analytics in Industrial IoT Using a Concentric Computing Model. IEEE Communications Magazine **56**, 37–43 (2018)
8. Peng, S., Wansen, F., Ruobing, H., Shengen, Y., Yonggang, W.: Optimizing Network Performance for Distributed DNN Training on GPU Clusters: ImageNet/AlexNet Training in 1.5 Minutes. IEEE Transactions on Big Data (2019)
9. Konys, A.: An Ontology-Based Knowledge Modelling for a Sustainability Assessment Domain. Sustainability **10**(2) (2018)
10. Schmidt, B., Wang, L., Galar, D.: Semantic Framework for Predictive Maintenance in a Cloud Environment. In: Teti, R., Doriana, M., Addona, D. (eds) 10th CIRP Conference on Intelligent Computation in Manufacturing Engineering - CIRP ICME 16, LNCS, vol. 62, pp 583-588. Elsevier
11. Lira, N. D., Borsato, M.: OntoProg: An ontology-based model for implementing Prognostics Health Management in mechanical machines. Advanced Engineering Informatics **38**, 746–759 (2018)
12. Xu, F., Liu, X., Chen, W., Zhou, C., Cao, B.: Ontology-Based Method for Fault Diagnosis of Loaders. Sensors **18**(3) (2018)
13. Cao, Q., Giustozzi, F., Zanni, M. C., De, B., De, B. F., Reich, C.: Smart Condition Monitoring for Industry 4.0 Manufacturing Processes: An Ontology-Based Approach. Cybernetics and Systems **50**, 1–15 (2019)
14. Evgeny, K., Gulnar, M., Ognjen, S., Guohui, X., Elem, G. K., Mikhail, R.: Semantically-enhanced rule-based diagnostics for industrial Internet of Things: The SDRL language and case study for Siemens trains and turbines. Journal of Web Semantics **56**, 11–29 (2019)
15. Toufik, B., Benidir, M.: Bearing faults diagnosis using fuzzy expert system relying on an Improved Range Overlaps and Similarity method. Expert Systems with Applications **108** (2018)
16. Antomarioni, S., Pisacane, O., Potena, D., Bevilacqua, M., Ciarapica, F. E., Diamantini, C.: A predictive association rule-based maintenance policy to minimize the probability of breakages: application to an oil refinery. The International Journal of Advanced Manufacturing Technology **105** (2019)

17. Bernard, G.: Rule mining in maintenance: Analysing large knowledge bases. Computers & Industrial Engineering **139** (2020)
18. Zhang, L., Zhongqiang, M., Sun, C. Y.: Remaining Useful Life Prediction for Lithium-Ion Batteries Based on Exponential Model and Particle Filter. IEEE Access (2018)
19. Sevegnani, M., Calder, M.: Stochastic Model Checking for Predicting Component Failures and Service Availability. IEEE Transactions on Dependable and Secure Computing (2017)
20. Aizpurua, U. J., Catterson, V., Abdulhadi, I., Segovia, M.: A Model-Based Hybrid Approach for Circuit Breaker Prognostics Encompassing Dynamic Reliability and Uncertainty. IEEE Transactions on Systems, Man, and Cybernetics: Systems **48**(9), 1–12 (2017)
21. Nan, C., Zhi, S. Y., Yisha, X., Linmiao, Z.: Condition-based maintenance using the inverse Gaussian degradation model. European Journal of Operational Research **243**(1), 190–199 (2015)
22. Donghui, P., Jia, B. L., Jinde, C.: Remaining useful life estimation using an inverse Gaussian degradation model. Neurocomputing **185**, 64–72. (2016)
23. Vianna, W., Yoneyama, T.: Predictive Maintenance Optimization for Aircraft Redundant Systems Subjected to Multiple Wear Profiles. IEEE Systems Journal **12**(2),1–12 (2017)
24. Nathalie, C., Karel, M., Aless, , Ro, A.: Model-based predictive maintenance in building automation systems with user discomfort. Energy **138**, 306–315 (2017)
25. Keizer, M., Flapper, S., Teunter, R.: Condition-based maintenance policies for systems with multiple dependent components: A review. European Journal of Operational Research **261** (2017)
26. Wuest, T., Weimer, D., Irgens, C., Thoben, K. D.: Machine learning in manufacturing: Advantages, challenges, and applications. Production & Manufacturing Research **4**, 23–45 (2016)
27. Teng, W., Zhang, X., Liu, Y., Ma, Z.: Prognosis of the Remaining Useful Life of Bearings in a Wind Turbine Gearbox. Energies **10**(32) (2016)
28. Karmacharya, I., Gokaraju, R.: Fault Location in Ungrounded Photovoltaic System Using Wavelets and ANN. IEEE Transactions on Power Delivery **33**(2), 549–559 (2017)
29. Netam, G., Yadav, A.: Fault Detection, Classification and Section Identification in Distribution network with D-STATCOM using ANN. International Journal of Advanced Technology and Engineering Exploration **4** (2016)
30. Chine, W., Mellit, A., Lughi, V., Malek, A., Sulligoi, G., Massi, P. A., Ro, : A novel fault diagnosis technique for photovoltaic systems based on artificial neural networks. Renewable Energy **90**, 501–512 (2016)
31. Abdallah, I., Dertimanis, V., Mylonas, C., Tatsis, K., Chatzi, E., Dervilis, N., Worden, K., Maguire, A.: Fault diagnosis of wind turbine structures using decision tree learning algorithms with big data. Safety and Reliability - Safe Societies in a Changing World (2018)
32. Rabah, B., Samir, M.: Fault detection and diagnosis based on C4.5 decision tree algorithm for grid connected PV system. Solar Energy **173**, 610–634 (2018)
33. Sangram, P., V, M. P.: Fault Detection of Anti-friction Bearing using Ensemble Machine Learning Methods. International Journal of Engineering, Transactions B: Applications **31**, 1972–1981 (2018)
34. Han, H., Cui, X., Fan, Y., Qing, H.: Least squares support vector machine (LS-SVM)-based chiller fault diagnosis using fault indicative features. Applied Thermal Engineering **154**, 540–547 (2019)

35. Zhu, X., Xiong, J.: Fault Diagnosis of Rotation Machinery Based on Support Vector Machine Optimized by Quantum Genetic Algorithm. IEEE Access **6**, 33583–33588 (2018)
36. Liu, Z., Mei, W., Zeng, X., Yang, C., Zhou, X.:Remaining useful life estimation of insulated gate biploar transistors (igbts) based on a novel volterra k-nearest neighbor optimally pruned extreme learning machine (vkopp) model using degradation data. Sensors, **7**(11) 2524 (2017)
37. Qin, H., Xu, K., Ren, L.: Rolling Bearings Fault Diagnosis via 1D Convolution Networks. In: 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP), pp. 617–621. IEEE, Wuxi, China (2019)
38. Liu, X., Zhou, Q., Shen, H.: Real-time Fault Diagnosis of Rotating Machinery Using 1-D Convolutional Neural Network. In: 2018 5th International Conference on Soft Computing & Machine Intelligence (ISCMI), pp. 104-108. IEEE, Nairobi, Kenya (2018)
39. Kiranyaz, S., Gastli, A., Ben, B. L., Alemadi, N., Gabbouj, M.: Real-Time Fault Detection and Identification for MMC Using 1-D Convolutional Neural Networks. IEEE Transactions on Industrial Electronics **66**(11), 8760–8771(2018)
40. Ren, L., Sun, Y., Wang, H., Zhang, L.: Prediction of Bearing Remaining Useful Life With Deep Convolution Neural Network. IEEE Access **6**, 13041 - 13049 (2018)
41. Babu, G., Zhao, P., Li, X.: Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life. In: Navathe, S., Wu, W., Shekhar, S., Du, X., Wang, X., Xiong, H. (eds) Database Systems for Advanced Applications. DASFAA 2016. LNCS, vol 9642. Springer, Cham. https://doi.org/10.1007/978-3-319-32025-0 (2016)
42. Xingqiu, L., Jiang, H., Hu, Y., Xiong, X.: Intelligent Fault Diagnosis of Rotating Machinery Based on Deep Recurrent Neural Network. In: 2018 International Conference on Sensing,Diagnostics, Prognostics, and Control (SDPC), pp. 67–72. IEEE, Xi'an, China (2018)
43. Yuan, J., Tian, Y.: An Intelligent Fault Diagnosis Method Using GRU Neural Network towards Sequential Data in Dynamic Processes. Processes **7**(3) (2019)
44. Yang, R., Huang, M., Lu, Q., Zhong, M.: Rotating Machinery Fault Diagnosis Using Long-short-term Memory Recurrent Neural Network. IFAC-PapersOnLine **51**, 228-232 (2018)
45. Chen, J., Jing, H., Chang, Y., Liu, Q.: Gated Recurrent Unit Based Recurrent Neural Network for Remaining Useful Life Prediction of Nonlinear Deterioration Process. Reliability Engineering & System Safety **185**, 372–382 (2019)
46. Hong, J., Wang, Z., Yao, Y.: Fault prognosis of battery system based on accurate voltage abnormity prognosis using long short-term memory neural networks. Applied Energy **251** (2019)
47. Wu, Q., Ding, K., Huang, B.: Approach for fault prognosis using recurrent neural network. Journal of Intelligent Manufacturing **31** (2020)
48. Akcay, S., Atapour, A. A., Breckon, T.: GANomaly: Semi-supervised Anomaly Detection via Adversarial Training. In: Jawahar C., Li H., Mori G., Schindler K. (eds) Computer Vision – ACCV 2018. ACCV 2018. LNCS, vol 11363. Springer, Cham. https://doi.org/10.1007/978-3-030-20893-6
49. Jiang, W., Hong, Y., Zhou, B., He, X., Cheng, C.: A GAN-Based Anomaly Detection Approach for Imbalanced Industrial Time Series. IEEE Access **7**, 143608–143619 (2019)
50. Khan, S., Prosvirin, A., Er, , Kim, J.: Towards Bearing Health Prognosis using Generative Adversarial Networks: Modeling Bearing Degradation. In: 2018 Inter-

national Conference on Advancements in Computational Sciences (ICACS). IEEE, Lahore, Pakistan (2018)
51. Chicco, D.: Siamese Neural Networks: An Overview. n: Cartwright H. (eds) Artificial Neural Networks. Methods in Molecular Biology, vol 2190. Humana, New York, NY.
52. Gregory, R. K.: Siamese Neural Networks for One-Shot Image Recognition. ICML deep learning workshop **2**, (2015)
53. Pei, W. and T., Van Der Maaten, D. and L.: Modeling Time Series Similarity with Siamese Recurrent Networks. ArXiv preprint. arXiv:1603.04713. (2016)
54. Klein, P., Weingarz, N., Bergmann, R.: Enhancing Siamese Neural Networks through Expert Knowledge for Predictive Maintenance. In: Gama J. et al. (eds) IoT Streams for Data-Driven Predictive Maintenance and IoT, Edge, and Mobile for Embedded Machine Learning. ITEM 2020, IoT Streams 2020. Communications in Computer and Information Science, vol 1325. Springer, Cham. https://doi.org/10.1007/978-3-030-66770- (2020)
55. Google Cloud AutoML Tables Documentation, https://cloud.google.com/automl-tables/docs. Last Accessed 22 September 2021
56. Google Cloud BigQuery ML Documentation, https://cloud.google.com/bigquery-ml/docs. Last accessed 22 September 2021
57. Zenodo - predictive maintenance dataset, https://doi.org/10.5281/zenodo.3653909. Last accessed 21 September 2021
58. Arias, C. M., Kulkarni, C., Goebel, K., Fink, O.: Aircraft Engine Run-to-Failure Dataset under Real Flight Conditions for Prognostics and Diagnostics. NASA Ames Prognostics Data Repository, NASA Ames Research Center **6**(5) (2021)
59. Wang, B., Lei, Y., Li, N., Li, N.: A Hybrid Prognostics Approach for Estimating Remaining Useful Life of Rolling Element Bearings. IEEE Transactions on Reliability **69**, 1-12 (2018)
60. Google Cloud Vertex AI, https://cloud.google.com/vertex-ai. Last accessed 22 September 2021
61. Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J.: Google Vizier: A Service for Black-Box Optimization. In: proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 17), pp. 1487–1495. Association for Computing Machinery, New York, NY, USA (2017)