POLITECNICO DI TORINO
Repository ISTITUZIONALE

A robust methodology for dataset preparation and algorithm performance assessment in machine learning prediction of the fatigue life of additive manufactured components

Original
A robust methodology for dataset preparation and algorithm performance assessment in machine learning prediction of the fatigue life of additive manufactured components / Di Maggio, L.G., Gastaldi, C., Renzo, D.A., Delprete, C., Furgiuele, F.. - In: ENGINEERING WITH COMPUTERS. - ISSN 0177-0667. - 41:5(2025), pp. 2937-2952. [10.1007/s00366-025-02139-7]

Availability:
This version is available at: 11583/2999417 since: 2025-06-18T11:00:18Z

Publisher:
Springer

Published
DOI:10.1007/s00366-025-02139-7

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

28 November 2026

Engineering with Computers (2025) 41:2937–2952
https://doi.org/10.1007/s00366-025-02139-7

ORIGINAL ARTICLE

A robust methodology for dataset preparation and algorithm performance assessment in machine learning prediction of the fatigue life of additive manufactured components

Luigi Gianpio Di Maggio[1] · Chiara Gastaldi[1] · Danilo Antonello Renzo[2] · Cristiana Delprete[1] · Franco Furgiuele[3]

Received: 2 July 2024 / Accepted: 25 March 2025 / Published online: 18 April 2025
© The Author(s) 2025

Abstract
The paper leverages existing literature on fatigue life prediction in additive manufacturing components within the realm of artificial intelligence (AI) and machine learning methods. The first key contribution is a meticulous methodology for dataset preparation. It aims to convert features from natural language to a format suitable for AI algorithms, minimizing over-fitting risks. This process is not limited to AI-specific scalers, but leverages authors' expertise in the problem's physics. Additionally, the article evaluates the performance of various machine learning and AI algorithms. This constitutes the second contribution of the paper. Typically, the performance of different algorithms is assessed based on the root mean square comparison of predicted versus true data. However, this widely accepted methodology overlooks the fact that data points are not independent entities; rather, they are grouped into subsets, each associated with a specific fatigue Wöhler curve. From a practical standpoint, the engineering entity of interest for prediction is the complete curve, not just the single point belonging to the curve. Therefore, a methodology based on this concept is proposed to reliably assess algorithm performance, serving as a complementary technique to standard performance assessment metrics.

Keywords Fatigue · Additive manufacturing · Machine learning · Fatigue life · Fatigue dataset · High cycle fatigue

1 Introduction

Additive manufacturing (AM) is the most popular manufacturing process in Industry 4.0, which enables on-demand production, unlocks digital design tools, and offers breakthrough performance and high flexibility. Other advantages consist in the manufacturing of components with complex geometry and with a minor amount of material waste and energy consumption, with respect to the subtractive manufacturing processes. The use of AM parts in the engineering fields is limited due to the difference and uncertainty observed in the mechanical properties of AM components compared to traditionally manufactured ones [1, 2]. This uncertainty is especially detrimental when considering the fatigue behavior, as it directly impacts the capability of effectively predicting the life of AM components. Such uncertainty is mainly due to two contributing factors:

- large number of process parameters, each of which may impact the mechanical properties;
- lack of control on process parameters, resulting in lack of repeatability in the manufacturing process.

✉ Chiara Gastaldi
chiara.gastaldi@polito.it

Luigi Gianpio Di Maggio
luigi.dimaggio@polito.it

Danilo Antonello Renzo
daniloantonello.renzo@uniroma1.it

Cristiana Delprete
cristiana.delprete@polito.it

Franco Furgiuele
franco.furgiuele@unical.it

[1] Dipartimento di Ingegneria Meccanica e Aerospaziale, Politecnico di Torino, Corso Duca degli Abruzzi, 10129 Turin, Italy

[2] Department Chemical Engineering Materials Environment, Sapienza University of Rome, Via Eudossiana 18, 00184 Roma, Italy

[3] Department of Mechanical, Energy and Management Engineering, University of Calabria, Via P. Bucci 46C, 87036 Rende, Italy

While the second factor is connected to technology and beyond the scope of the present work, the first one has been the focus of recent scientific research in the field [3, 4]. In detail, different authors have evaluated the influence of AM process parameters such as scan speed, energy beam power, hatch space, layer thickness, surface roughness, yield strength, ultimate strength, elongation, stress amplitude, etc. [5, 6]. Other relevant parameters, not directly connected to AM, are the processing sequence of heat treatment, Hot Isostatic Pressing (HIP) treatment, fatigue test temperature, fatigue test environment, porosity, residual stresses, etc. By taking action on these parameters, it is feasible to enhance the strength of these materials. Nevertheless, the limited understanding of this emerging technology continues to hinder its widespread implementation in the industrial sector. It is evident that developing models to accurately measure and forecast the impact of these characteristics on the final product's properties is a challenging task, especially when it comes to estimating fatigue strength, which is inherently stochastic.

In recent years, thanks to the increased availability of digitalized data, machine learning (ML) techniques have started to play an important role in materials research [7]. ML is a programming technique where a system learns from data to make predictions or classifications automatically. Namely, the statistical interrelation occurring between engineering inputs and outputs does not necessarily need to be represented by physical models, which might be otherwise limited by our capability to model physical phenomena. Instead, links and relations can be accurately derived by resorting to data-driven models, including in particular ML techniques. This involves uncovering the relationships between labeled data points within a specific training dataset, then the trained models are tested under new unlabeled data.

Over the past decades, several ML models and algorithms were applied to carry out the prediction of the fatigue life of AMed components. Indeed, ML algorithms such as Super Vector Machine (SVM), eXtreme Gradient Boosting (XGBoost), Linear Regression (LR), Random Forest (RF), Artificial Neural Networks (ANN), and Deep Neural Networks (DNN) have been employed to predict the fatigue life for different AM alloys [5, 8–11]. Zhang et al. [8] constructed two models utilizing distinct input features, employing 139 S-N data points obtained from experimental research on 316 L stainless steel manufactured through laser powder bed fusion (L-PBF). The first model utilizes process parameters and stress state, whereas the second model integrates tensile and elongation properties in addition to the stress state. Linguistic labels are employed to apply adaptive neuro-fuzzy inference to the model. The model evaluation is performed using cross-validation and the root mean square error (RMSE) serves as the criterion for error assessment. Balamurugan et al. [12] developed a data-driven approach for fatigue prediction of Ti-6Al-4V parts fabricated by laser powder bed fusion, combining a classification model with a Probabilistic Physics-Guided Neural Network for accurate and physically consistent predictions. Gao etl al. [13] introduced a novel neuro-fuzzy-based machine learning method for predicting multiaxial fatigue life of metallic materials by combining fuzzy inference and neural networks. Zhu et al. [14] performed high cycle fatigue life prediction of titanium alloys based on a deep learning approach incorporating advanced attention mechanism and explored key factors influencing HCF life by using Shapley value. Luo et al. [9] adopted mean absolute error (MAE), mean squared error (MSE), and the coefficient of determination R^2 as evaluation metrics to assess several ML models for Inconel 718 produced using selective laser melting (SLM). These metrics were generated on the global dataset. In addition, the authors augmented the dataset by using data from prior studies to strengthen the reliability of the model. He et al. [10] evaluated individual S-N curves and compared the predicted values with the actual values. They employed overall evaluation criteria, such as global *RMSE* and life factor, to assess ML models. Bao et al. [11] employed X-ray tomography to analyze defect characteristics and develop Support Vector Machine (SVM) models for Ti-6Al-4V produced by SLM. There is no mention of specific S-N curves, and the assessment metrics are derived on a global scale due to the small size of the dataset. Also, the effect of manufacturing defects on fatigue life of selective laser melted Ti-6Al-4V structures was investigated. For instance, Han et al. [15] investigated the effect of defects on fatigue life prediction of additive manufactured Ti-6Al-4V by using micro-computed tomography and fracture-mechanics-based simulation, Markham et al. [16] investigated mixed-mode small fatigue crack growth rates focusing on Ti-6Al-4 V and 17-4 PH stainless steel and Li et al. [17] proposed physics-informed neural network framework based on fatigue indicator parameters for very high cycle fatigue life prediction of additively manufactured titanium alloy.

While there is a significant amount of scientific literature on the topic, existing research primarily focus on datasets that are limited to specific materials, process parameters, stress conditions, or include comparable data points from similar tests. The limitation is mostly due to the inherent experimental challenges involved in obtaining large amounts of data that investigate a wide range of scenarios for AMed materials. In light of this, Zhang and Xu [18] conducted an extensive review of the existing experimental literature on the subject to obtain a comprehensive dataset. In order to accomplish this, the authors utilized automated literature search methods, text categorization, as well as information retrieval from images and tables and compiled a comprehensive dataset that represents the existing experimental research on the topic. However, the existing literature shows

scant evidence of work investigating ML modeling on wide-ranging datasets that portray general distributions regarding fatigue of AMed materials. One of the goals of the present paper is to leverage the general dataset proposed in [18], rather than focusing on specific datasets, to the purpose of gauging the influence of different processing parameters and stress conditions on AMed alloys.

Furthermore, the pertinent literature frequently discusses model evaluation metrics that are comprehensive and assessed on randomly selected data points, even from distinct S-N curves. This approach aligns with ML methodologies, but it fails to consider metrics that evaluate the models' abilities to operate on individual S-N curves or predict new S-N curves whose points are not present in the training dataset. For this reason, the authors of this paper claim that an approach solely based on metrics that do not take into account the stratification of the dataset into different fatigue curves may not be reliable enough to estimate the capabilities of predictive models in real-world contexts and applications. In this regard, it is beneficial to assess metrics that apply to full S-N curves in addition to those frequently employed and related to points randomly extracted from different curves. Then, the second contribution of this paper is related to model evaluation, specifically which metric is best suited to assess the performance of ML algorithms in the context of fatigue data prediction. In detail, two different techniques are used, and their performances are compared in Sect. 3. On one hand, a state-of-the-art cross-validation technique based on the *RMSE* is used. On the other hand, a second metric is proposed to take into account the fact that, from a practical engineering point of view, data are grouped into single material curves. It is argued that the performance of an algorithm should be assessed on its capability to predict an entire fatigue strength curve not used during training (rather than a single point). This metric constitutes, in the authors' opinion, a necessary indicator of whether ML algorithms can partially replace experimental campaigns. Namely, in this study, the authors propose the utilization of the Weighted Average Percentage Error (*WAPE*) metric to assess the prediction capabilities of machine learning on complete S-N curves. By employing this statistic, the authors find that the evaluation of algorithms can diverge from the assessment based solely on global metrics. The robustness and reliability of the methodology also comes from the use of cross-validation, which reduces the risk that a randomly chosen test set is not representative of the actual operating conditions of the model. Through this approach, all data are alternately used for training and validation, ensuring that the reported performance is more indicative of that expected on real data. Moreover, robustness is also understood as the ability to avoid overfitting, ensuring that the model does not overfit the training data.

The paper is organized as follows. Section 1 introduces the topic of material fatigue and ML methodologies in this field, stating the objectives of the work. Section 2 recounts the methodology used to apply ML algorithms to fatigue data of AM components. Moreover, the section presents dataset preparation technique that ensures an optimized data format to enhance the performance of ML algorithms while maintaining a strong connection to the underlying physical phenomena, thereby reducing the risk of over-fitting. Section 3 presents the results and discusses the implications related to the use of metrics and the prediction of completely unseen S-N curves. Finally, conclusions are drawn in Sect. 4.

2 Methodology

The methodology employed in this study involves a series of processes designed to render the data suitable for the application of ML models. The process of feature selection and feature transformation is implemented in conjunction with data cleaning. Similarly, the process of standardizing data and encoding categorical features is essential for effective implementation. Once the target variable was identified, various models were subsequently evaluated, ensuring that they were cross-validated. Namely, the research aims to leverage wide-ranging fatigue data from AMed samples found in the literature [18] and apply ML algorithms to construct a predictive framework capable of linking relevant input variables to the fatigue behaviour. Input features available in the literature are diverse and heterogeneous across different sources. Drawing on the well-known literature on fatigue phenomena, this work eliminates non-relevant features while retaining those proven to influence fatigue behavior. These include the AM process type, heat treatment processing sequence, surface finish characteristics, fatigue test parameters, and material mechanical properties. This section outlines the methodology employed in this study.

2.1 Dataset preparation

The dataset [18], depicted in Fig. 1, reported the stress amplitude σ_a, versus the cycles of failure N_f, and collects more than 1500 fatigue sub-datasets for different AM alloys. The starting dimension of the mined dataset is [15,146 × 33], i.e. 15,146 data points characterized by 33 features. The dataset, first structured in [18], is not optimized for ML applications, therefore a dataset preparation phase is necessary.

The common tasks steps for data preparation are:

- Feature selection. Taking advantage of the authors' knowledge of the physical phenomenon, further supported by relevant literature [19, 20], only the features

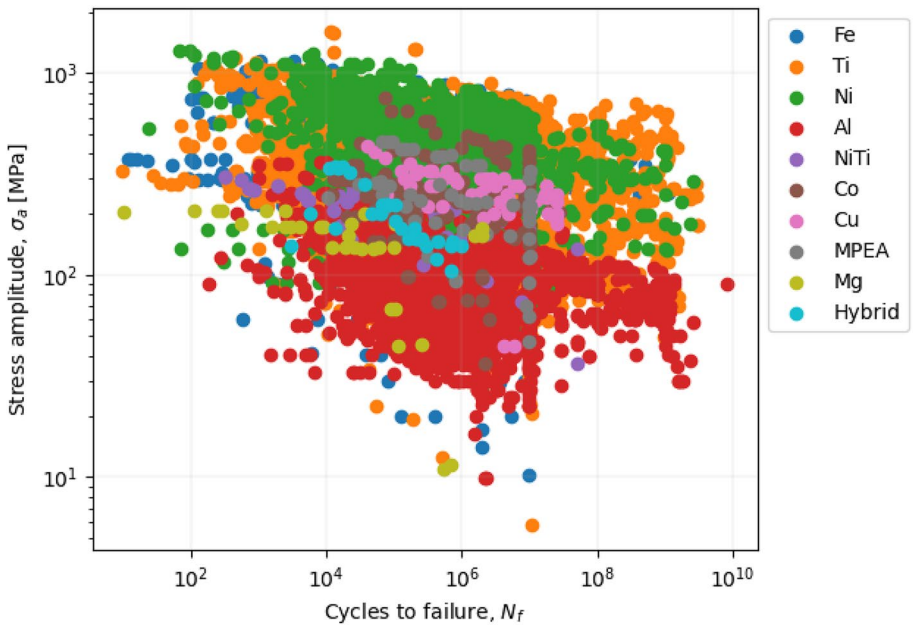

Fig. 1 Data points represented on the life-alternating stress plane

that are truly relevant to fatigue life prediction have been included. Others, useful to identify the specific experimental campaign, but irrelevant to fatigue life prediction (e.g. frequency and load control of the fatigue test), have been removed to reduce the number of input variables. This step is important because irrelevant and redundant input variables can distract or mislead learning algorithms. In addition, the columns (i.e. features) that had a single data observation or value were useless for modelling and they were removed. These columns are referred to as zero-variance predictors because the measured variance is zero. Ultimately, the information about the "Processing sequence and parameters" characteristics was divided into eight detailed features: hot isostatic pressing (HIP) treatment, temperature HIP, pressure HIP, time of HIP, heat treatment (HT), maximum temperature of HT, time of HT, and surface finish, for which the categorical information was converted into mean roughness values. Table 1 details the features which have been deemed significant for fatigue strength/life prediction.

- Data cleaning. Once the relevant features have been selected, it was here decided to remove all data rows with missing features along with rows with duplicate data. Duplicate data are not useful for the modeling process and can be misleading during model validation. In the literature, there are options, known as data imputation [21–23], to provide incomplete data with an estimate of the missing features. It was here decided not to take advantage of such techniques, to avoid introducing bias during the training process. However, models with varying degrees of significance could be constructed by incorporating the missing rows after purposely developed data imputation technique has been performed. The final size of the dataset was [1184 × 27] [24–43].
- After the initial steps of data cleaning (Fig. 2), the subsequent stages of data preparation involve encoding and standardization. Data encoding is crucial in machine learning as it transforms categorical data into numerical format, which algorithms can easily interpret. In the present case, the ordinal nature of the Label Encoder method could potentially cause issues for some algorithms. However, none of the models showed significant performance degradation. Additionally, models such as Random Forests and XGBoost are not sensitive to the artificial hierarchies introduced by Label Encoding, as they split data based on feature values rather than relying on distances or ordinal relationships label encoding was used, assigning a unique integer value to each categorical data. It was decided not to use one-hot encoding in the given scope. This decision was made because using one-hot encoding for certain features with multiple possible manifestations could result in a significant increase in the number of columns in the dataset. This increase in columns could lead to overfitting issues, especially when there is a small amount of data available. Also, ML models generally perform better when input variables are scaled to a standard range of values. To achieve this, we normalized the dataset on the base of the standard score. This technique scales each input variable independently by subtracting the mean (centering) and dividing by the standard deviation, resulting in a distribution with a mean of zero and a standard deviation of one.

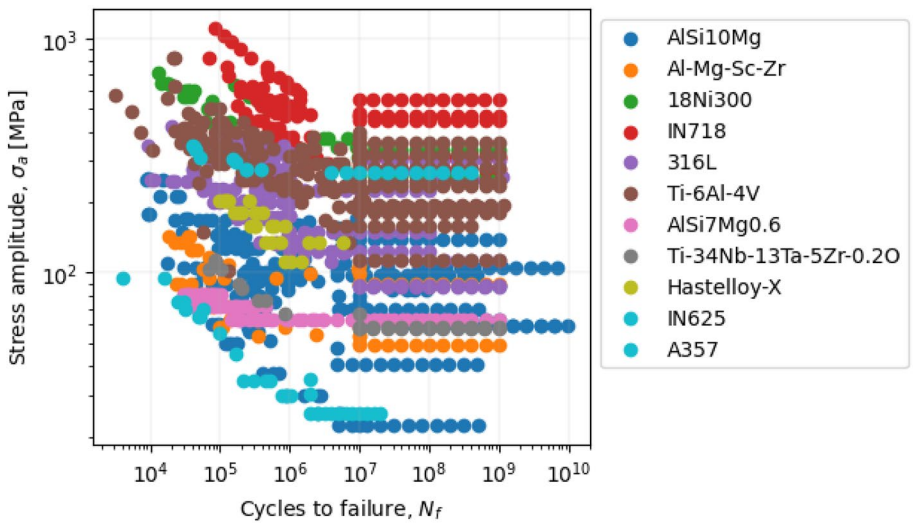

Fig. 2 Stress-life data points for different AM alloy that remain after data cleaning

Table 1 Features before and after data preprocessing

Original feature	New feature	Data type
Material		Categorical
Types of AM process		Categorical
Beam power energy		Numerical
Scan speed		Numerical
Hatch space		Numerical
Layer thickness		Numerical
Specimen growth direction		Numerical
Processing sequence and parameters	HIP treatment	Categorical
	Temperature HIP	Numerical
	Pressure HIP	Numerical
	Time HIP	Numerical
	Heat treatment (HT)	Categorical
	Maximum temperature HT	Numerical
	Time HT	Numerical
	Surface roughness	Numerical
Type of fatigue test		Categorical
Temperature of fatigue test		Numerical
Load ratio		Numerical
Specimen description		Categorical
Critical cross-section area		Numerical
Elastic modulus		Numerical
Yield strength		Numerical
Ultimate strength		Numerical
Elongation		Numerical
Cycles to failure		Numerical

2.2 Identification of the target variable

At this stage of the analysis, it becomes crucial to determine the target variable. Typically, in fatigue analysis, the technical diagram illustrates the fatigue limit (σ_a) plotted against the number of cycles (N) needed to induce specimen fatigue failure (N_f), as depicted in Fig. 2. Thus, it is logical to anticipate that the target variable will be either σ_a or N_f. Since metallic materials exhibit infinite life, characterized by a σ_a threshold below which no fatigue failure is expected to occur, this characteristic is evident in the dataset where some experimental points are designated as run-outs. In such

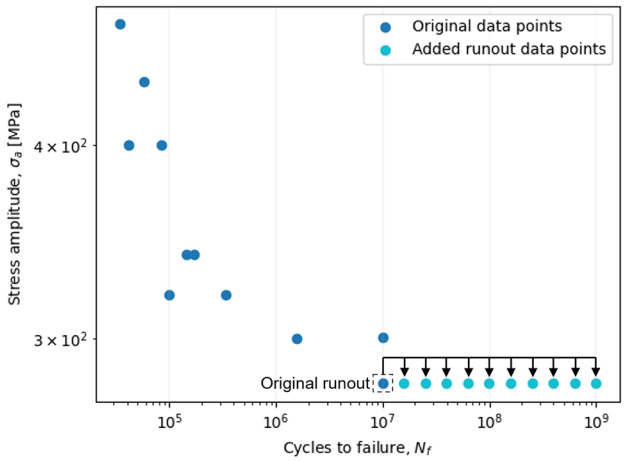

Fig. 3 Data points represented on the life-alternating stress plane

cases, no failure is observed, and the fatigue test is terminated after a specific number of cycles. Consequently, the corresponding fatigue limit σ_a remains valid for any N equal to or greater than the value associated with the run-out entry. As multiple values of N can correspond to the same σ_a, it was decided to select σ_a as the target variable. Also, later in the paragraph 2.5, a statistical rationale will be introduced to justify the inclusion of N as an independent variable. Additionally, the categorical binary feature representing the occurrence of run-out was eliminated, and instead, ten synthetic points with identical characteristics but increasing values of N were introduced. This adjustment, graphically represented in Fig. 3, not only reduced the number of features but also effectively conveyed the concept of infinite life to the algorithms.

2.3 Machine learning techniques

The machine learning approaches chosen for this research were evaluated based on their ability to reliably forecast fatigue life in different AM metals. This choice was informed by previous studies that identified ML algorithms best suited for this purpose [8, 9, 20, 44–46]. The algorithms commonly utilized for this purpose include Support Vector Regression (SVR), Kernel Ridge Regression (KRR), Random Forest Regression (RF), XGBoost, k-Nearest Neighbors (kNN), and Artificial Neural Networks (ANN). A brief overview of these algorithms is provided in the following. A more in-depth description is provided in Appendix.

The algorithms described Appendix typically require the setting of characteristic hyperparameters. In this study, the default parameter values available in standard ML libraries were chosen. This decision aligns with the paper's primary objective of assessing the suitability of the algorithms for fatigue life prediction. While ad-hoc optimization could potentially enhance algorithm performance, it is not the primary focus of this article. Furthermore, the optimization of hyperparameters may necessitate the utilization of grid-search techniques or Bayesian approaches. During the first phase, these improvements may conflict with the overall purposes of this work, may require significant computational resources, and leading to biased comparisons between algorithms. The authors of this paper utilize ML models in their original form as they are readily available in common ML libraries. This means that no modifications or adjustments to hyperparameters are made. Nevertheless, the application of such methodologies is of fundamental importance in practical cases. The main topics related to usage libraries and hardware architectures are given in the Sect. 2.4.

2.4 Frameworks and hardware for implementing ML models

For the construction of ML algorithms, many open-source library, frameworks, and application programming interfaces (API) such as *scikit-learn*, *TensorFlow*, *Keras*, and *PyTorch* have been developed [7]. *Scikit-learn* is a *Python* library that integrates a wide range of ML algorithms. The framework *scikit-learn* is built on top of *SciPy* which is an ecosystem for mathematics, science, and engineering libraries: *numpy* which is the base package to create the N-dimensional array, *pandas* that is used for the manipulation and analysis of data, and *Matplotlib* which is a comprehensive library for creating 2D Plotting. *TensorFlow* is a software library for ML and AI. *TensorFlow* with the API *Keras* is a high-level programming language that provides a strong tool for the fast coding of ML and deep learning algorithms, mainly for industrial applications. *PyTorch* is a low-level programming language for developing and training neural networks and deep learning models. The low-level syntax involves a more verbose programming language which makes it more used in the research field.

The development of ML software also necessitated the development of more advanced computing technology. Indeed, a computer with a Central Processing Unit (CPU) could be not fast enough for the computing inference phase of ML projects that implement ANN algorithms. Moreover, even multi-core CPUs are not fast enough when processing large amounts of data. Therefore, accelerator computing devices composed of multiple Arithmetic Logic Units (ALUs) could improve computing performance. Two different types of these devices are Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs). GPUs were designed to handle massive amounts of parallel computing processes and have proven particularly effective in tasks involving image processing and convolutional neural networks (CNNs). However, they consume a lot of energy. On

the other hand, TPUs were developed specifically for neural network ML, providing high computing efficiency but with lower precision. The main advantage is that TPUs use less energy compared to GPUs. TPUs run on Application-Specific Integrated Circuits (ASICs) and are optimized for TensorFlow. The choice of hardware depends on the dimension of the dataset and model characteristics. For small datasets (i.e., fewer than 2000 rows) and less complex models, a modern CPU with multiple cores may be adequate. CPUs are adequate for ML applications that require more sequential or logical operations, such as decision trees. However, for larger datasets and complex models like Deep Neural Networks, the use of GPUs reduce significantly the training times. Cloud solutions such as Amazon Web Services (AWS), Google Colab, Google Cloud Platform (GCP), and Microsoft Azure provide access to computing resources, allowing users to choose the hardware device based on their needs and budget. Another important factor is the memory storage for the ML training. These can influence the speed and quality of the training. Therefore, is important to choose memory and storage that can accommodate the size and complexity of the dataset and ML model, frequency, and duration of training steps. In this study, the training and the algorithm assessment were conducted on a personal computer equipped with an Intel Core i7-8750H CPU with a base frequency of 2.20 GHz, (capable of 4.1 GHz with Turbo Boost used for the high workloads, such as the training phases of ML algorithms) and a 64-bit operating system. The Python environment was configured with scikit-learn version 1.7 and PyTorch version 2.5.1 for DNN algorithm, running on Windows 10. This local hardware configuration was chosen to simulate conditions for ML development in small-scale environments.

2.5 Models' evaluation

The models analyzed in the study were validated by means of *cross-validation* techniques. Cross-validation is a statistical method employed to evaluate the ability of a predictive model to generalize its outcomes. Cross-validation is beneficial in scenarios when the goal is to assess the practical performance expected by a predictive model. Also, this methodology is highly efficient and valuable when applied to predictive models utilized for data-driven decision-making [47]. The technique is executed in accordance with the following description.

1. Dataset partitioning. In the process of cross-validation, the dataset is partitioned into two or more distinct sets. The most prevalent approach is the *k*-fold method, which involves dividing the information into *k* groups, known as folds. For instance, in a 5-fold cross-validation, the dataset is partitioned into five distinct subsets.
2. During each iteration of cross-validation, a distinct fold is designated as the test set, while the remaining folds are combined and utilized as the training sets. The aforementioned procedure is iterated *k* times, with each fold being utilized precisely once as a test set.
3. Performance assessment. The model undergoes *k* iterations of training and testing, with the performance of the model being recorded in each iteration. Performances can be assessed using many indicators. For problems involving continuous variables, such as estimating the fatigue life of mechanical components [48] or determining the maximum allowable alternate stress, the most commonly used evaluation metrics are the mean square error (MSE), root mean square error (RMSE) and the coefficient of determination R^2 [11, 49, 50]. However, a variety of metrics exist.
4. Average performance. Ultimately, the model's performance across all *k* iterations is aggregated to yield a more reliable and unbiased evaluation of the ability to generalize.

It was observed that the dataset employed in this study contained fatigue tests with wide-ranging characteristics. Then, the variance in the number of cycles to failure (N_f), or their logarithmic values, was found to be significantly high. Even after normalization, this high variance could pose challenges for the training of machine learning algorithms. Consequently, the models were built by employing N_f as an input variable to predict the maximum allowable alternate stress. The calculation is performed using the logarithmic values. Two quality metrics were employed in this study for overall evaluation, namely the RMSE and R^2 reported respectively in Eqs. (1) and (2). The weighted average percentage error (WAPE) reported in Eq. (3), instead, is employed to assess the ability to predict entire S-N curves. The definitions are provided below, where n is the number of observations, y_i is the true value of the logarithmic alternate stress, $\hat{y}_i$ is the predicted logarithmic alternate stress, and $\bar{y}$ is the mean value of the true logarithmic alternate stresses.

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

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2}$$

$$WAPE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\sum_{i=1}^{n}|y_i|} \tag{3}$$

Cross-validation is crucial in building data-driven models. It mitigates selection bias by ensuring that each data

point in the dataset is utilized for both training and testing. This approach minimizes the possibility of the model being excessively optimized for a specific subset of data, enabling a fairer evaluation of its performance. Also, it offers a more reliable assessment of the model's behavior on new and unexplored data, which is essential for providing a comprehensive overview of the outcomes, thereby avoiding potential bias towards a specific scenario. This aids in drawing accurate and general conclusions. To assess the presence of over-fitting in cross-validated models, one can examine the performance variation over multiple folds, as there are more than just two sets used for training and testing. If the variation is small (i.e. small standard deviation of judgment metrics), it may be confidently stated that the model is not influenced by over-fitting. Cross-validation is further is applicable to nearly all forms of predictive models and can be utilized across a diverse array of fields. Nevertheless, it is important to emphasize that the computational effort can be consistent when dealing with extremely large deep learning models, making the application of cross-validation impractical.

This study employs a 5-fold cross-validation technique. Hence, the outcomes pertaining to actual and predicted maximum alternate stresses are showcased using a unified chart, as opposed to the conventional approach of employing two distinct graphs to exhibit training and testing phases. This comprehensive chart presents the aggregated results of all the data used as tests in the multiple cross-validation folds. This approach guarantees a comprehensive assessment of the model's performance on the full dataset, eliminating the necessity to distinguish between training and test outcomes. It is crucial to highlight that this does not imply that the training and test have not been conducted. Instead, five distinct training and test cycles were conducted, with each cycle corresponding to a different fold. Also, the charts display the results in the form of mean values and standard deviations, which are determined based on the outcomes of all k-folds. This yields a comprehensive statistical depiction of the model's performance, encompassing its ability to apply the acquired knowledge to fresh data during testing. By employing this approach, we can offer a general evaluation of the model's efficacy, considering its variability in performance across various subsets of data.

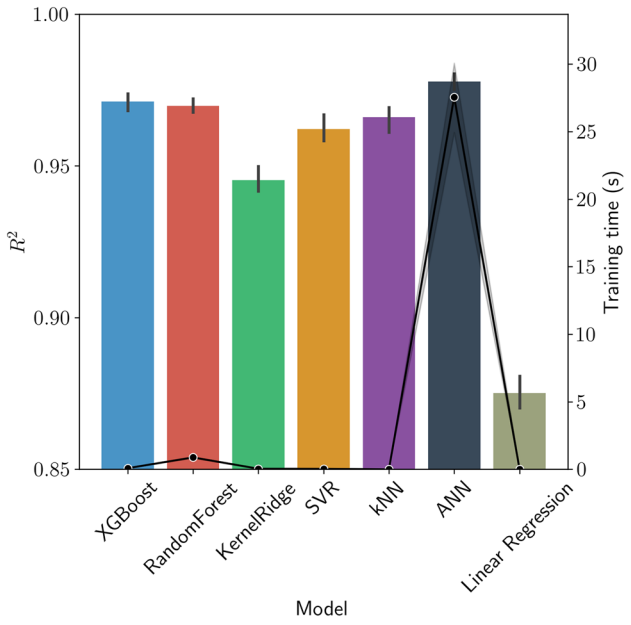

Fig. 4 R^2 scores and training times with 95% CI for cross-validated ML models

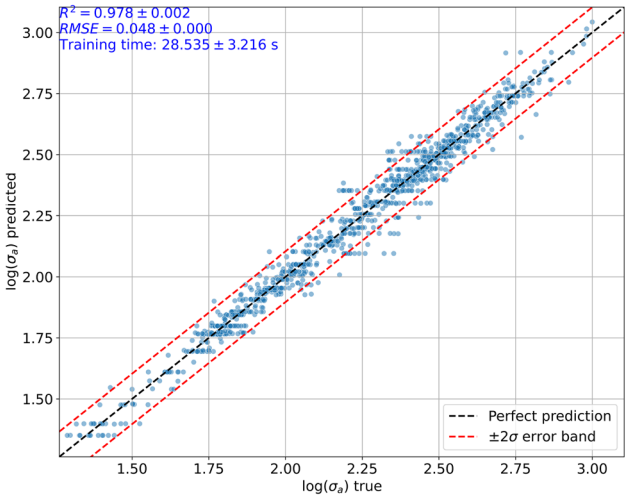

Fig. 5 True vs predicted stress amplitude by ANN

3 Results and discussion

The objective of this research is to assess the predictive capacities of ML algorithms in determining the performance of materials produced through AM under fatigue conditions. This is carried out by leveraging the general dataset proposed by Zhang and Xu [18]. Also, this work introduce the use of metrics that evaluate the models' abilities to operate on individual S-N curves or predict new S-N curves whose points are not present in the training dataset. Prior to analyzing the performance of predictive models for foreseeing fatigue strength curves, particularly those that are not present in the training dataset, it is essential to evaluate the capabilities of each algorithm by comparing their performance using the metrics and methodologies outlined in Sect. 2.5.

3.1 Algorithm comparison

Figure 4 shows the coefficients R^2 computed for the various models using cross-validation. More precisely, the bars depicted in the graph indicate the measured values for different folds, together with the 95% confidence interval (CI). Similarly, the black line refers to the axis of calculated training time for each model, with the 95% CI underlined by shaded area adjacent to the line. It is noteworthy that all the algorithms successfully achieve favorable R^2 values. Additionally, the CI width is rather narrow, indicating that the generated models with their parameters are less likely to suffer from overfitting in this particular scenario. The linear regression model was assessed to see whether the problem being analyzed requires the use of statistical learning and ML techniques, and is not readily solvable by linear regression models alone. The top-performing models include XGBoost, RF, and the ANN (Fig. 5). However, when taking computing time into account, the XGBoost model proves to be significantly more advantageous because of its capability of constructing several decision trees simultaneously, making it more efficient than the basic RF model. The accuracy of the ANN is influenced by the user's architecture design choices and the selection of specific hyperparameters, which can either accelerate or decelerate the training process, as well as impact the model's performances due to the peculiarities of the gradient descent method in its search for an optimal minimum.

3.2 Prediction of S-N data points

All previous ML algorithms (i.e., LR, SVR, KRR, RF, XGBoost, kNN, ANN) have been also used to predict all points of a stress-life curve (S-N curve). The complete development of these types of algorithms could become very useful in predicting a S-N data points for different types of AMed materials, avoiding carrying out experimental tests (or reducing these), and reducing the consumption of material and energy. The operating mechanism of these algorithms consists of not using for the training the subdataset that we want to predict (i.e. all S-N data points for an AMed material identified inside the dataset by a unique ID). Therefore, each algorithm was trained on a large amount of data because no split operation was applied, and at the end of the training each algorithm was asked to predict the stress amplitude for a given subset of data that contains the experimental cycles to failure with other information (ID not used for training). The prediction is considered good when the evaluated WAPE is less than 0.15 (Fig. 6). The authors made this choice because a WAPE of 0.15 is comparable to a condition where the entire predicted curve deviates by 15% from the true curve. This suggests that, on average, a curve with a WAPE of 0.15 is equivalent to a curve that is shifted by 15% from the true curve. However, we acknowledge that this approach does not provide a means to determine if the model estimate is cautious or too optimistic in relation to what may be observed in reality. The ML algorithms that are able to predict a major number of ID (i.e., with $WAPE < 0.15$) for different types of AM materials are KRR and XGBoost (Fig. 7). This finding is intriguing since when assessing the capability to forecast whole curves, a basic unoptimized ANN performs less effectively than a poorly performing KRR, which instead shows to be one of the top performers when evaluating WAPE. However, when using global judgment metrics like RMSE and R^2, KRR was found to be one of the lowest performers, second only to simple linear regression. RMSE and R^2 could be not suitable to assess the ability to forecast specific curves. However, the authors acknowledge that the findings achieved using global metrics and cross-validation are more robust from a statistical point of view with respect to the results concerning WAPE. This is due to the increased quantity of test data used and the averaging process employed in cross-validation. On the other hand, the fatigue curves used to calculate $WAPE$ have low numerosities. However, the statistical relevance of WAPE is inherent in the fact that it already contains information on several experimental-predicted points. The trends of the percentage of the predicted IDs using these two algorithms are shown in Fig. 8, and we can observe that for smaller WAPE (i.e., < 0.05) the XGBoost algorithm is able to predict a major number of IDs. Therefore, considering these last results (Figs. 4, 5, 6, 7) we can conclude that XGBoost is the ML algorithm that offers the best performance in terms of quality of prediction and training time.

These results highlight how the choice of metrics for evaluating predictive algorithms can influence the conclusions drawn from the analyses. However, an equally relevant aspect concerns the quality of the data, since it is well known that most fatigue data in the literature do not always follow recognized standards and do not necessarily include detailed information for design applications [51].

In spite of these critical issues widely discussed in the literature, our study shows that, based on a heterogeneous dataset, it is possible to generate fatigue curves that are completely absent in the training set, and not only partially so. Although this result is preliminary, it suggests some promising directions for further investigation. In parallel, the use of machine learning techniques can be further developed not only for the improvement of the predictive model, but also to implement strategies to evaluate and certify the quality of the data used, thus contributing to greater reliability of the predictions.

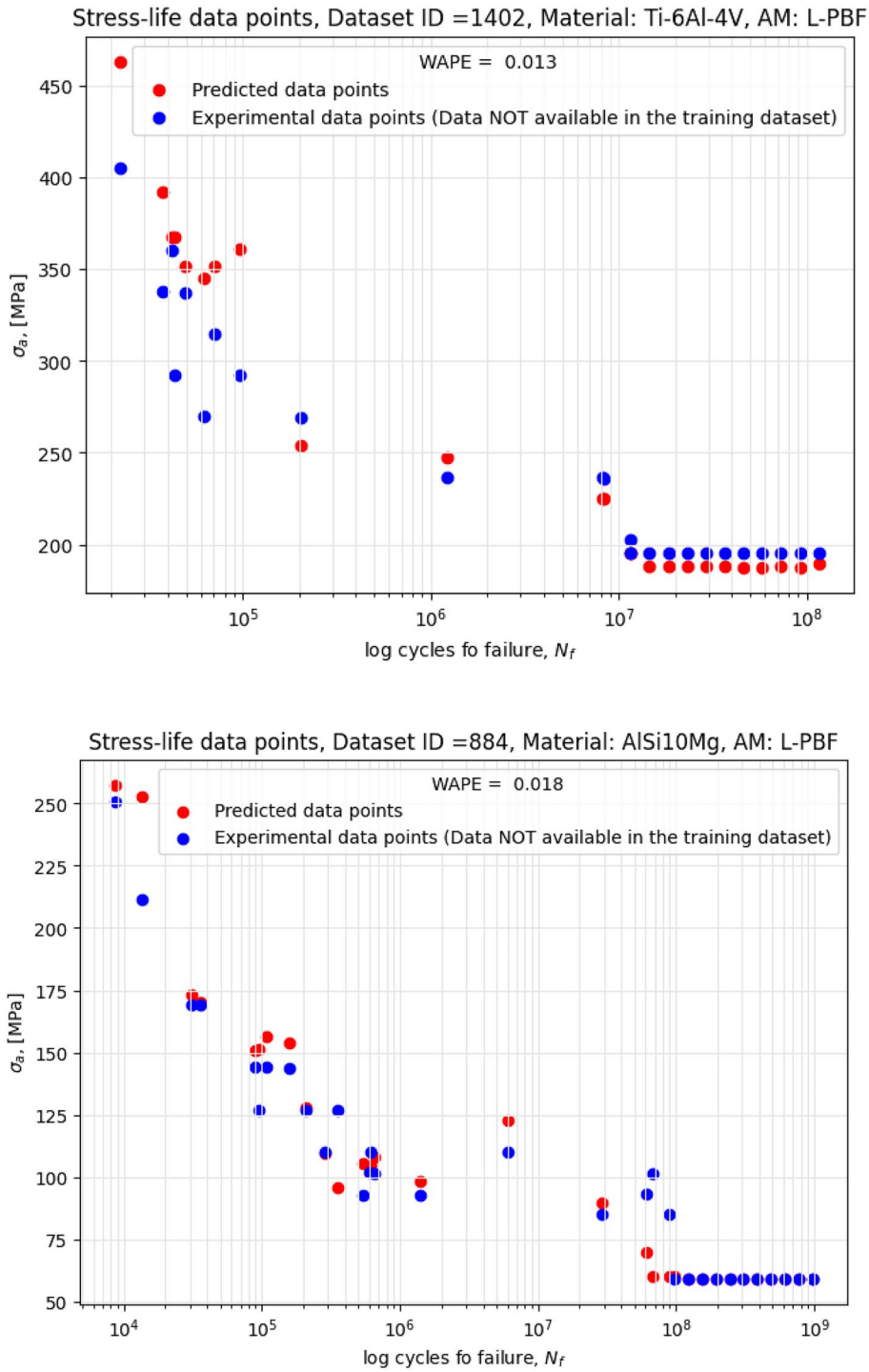

Fig. 6 Predicted data points vs experimental data points by XGBoost

4 Conclusions

This paper introduces a reliable approach for preparing datasets and evaluating the effectiveness of machine learning models in forecasting the fatigue life of additively manufactured components. The research centers on two primary contributions:

- the potentials of wide-ranging datasets were investigated and a procedure for preparing the dataset was proposed. Namely, features from natural language were transformed into inputs suitable for machine learning models. The strategy further mitigates the dangers of overfitting;
- the evaluation of performances is carried out by considering complete fatigue life curves instead besides individual data points. This methodology was enhanced

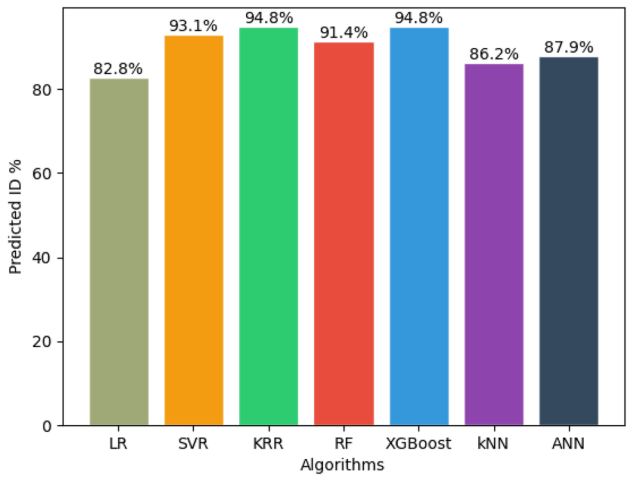

Fig. 7 Prediction of S-N data points for the different AMed material (WAPE < 0.15)

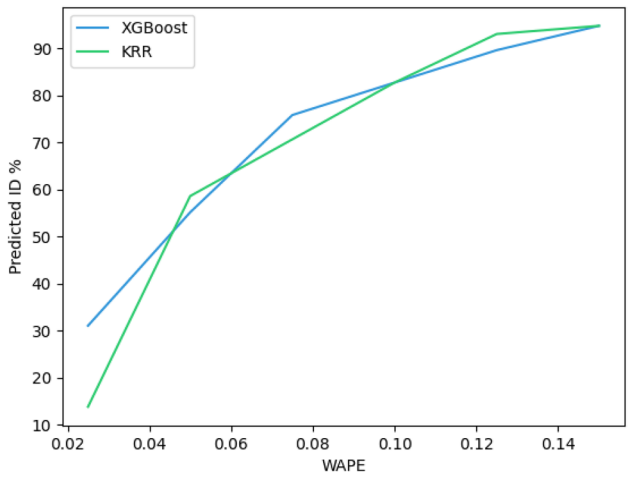

Fig. 8 Trend of predicted ID by KKR and XGBoost algorithms

by incorporating the weighted average percentage error (WAPE) as a statistic, which provides a more pragmatic engineering viewpoint compared to conventional methods like root mean square error (RMSE). The methodology is proposed as robust and reliable depending on the validation technique used and the metrics introduced to evaluate algorithms for their practical use.

Also, a wide range of machine learning methods was investigated to assess their efficacy in forecasting the fatigue life of different metals manufactured using additive manufacturing. XGBoost has proven to be the best performing model on all the judging metrics. Overall, this study provides insights to the field of materials science and engineering by suggesting approaches that can enhance the accuracy and feasibility of predicting fatigue life. These findings will encourage additional research and development in this field. Additional validations are required to develop models that can utilize the complete dataset and employ reliable methods for including missing values, especially those in categorical features. Additionally, it is necessary to deploy hyperparameter optimization strategies and conduct validation on external datasets. Moreover, further validations will be conducted using new experimentally obtained external datasets. This procedure will strengthen the conclusions of the present study and increase its generalizability.Such a validation process could also be aimed at analytically assessing the quality of the training data, using the variability of performance obtained on datasets from different sources as an indicator. To further improve the quality of the extracted data, it might be useful to explore more advanced extraction and filtering techniques to ensure greater consistency and reliability in the analyses. Then, future developments could include the use of language models not only for data collection but also to evaluate and interact with the knowledge and documents needed to develop the models developed in this paper.

Appendix: Machine learning algorithms

Linear regression algorithm

The linear regression algorithm finds the linear relationship between a dependent variable (i.e., target variable) and one or more independent features. In this last case, the method is known as multivariate linear regression. The equation for multiple linear regression is:

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n \quad (4)$$

where y is the dependent variable, X_n are the independent variables, β_0 is the intercept, β_1, β_2,... β_n are the slopes. The goal of the algorithm is to find the best-fit line that can predict the values based on the independent variables. The limit of this algorithm is that fails when there are too many independent variables to take into account. Linear regression continues to be among the most prevalent algorithms owing to its straightforwardness and ease of interpretation. Nevertheless, the efficacy of this model rely on foundational assumptions that may prove to be unsuitable for exceedingly intricate or nonlinear phenomena.

Random Forest algorithm

The Random Forest algorithm used in our case is a supervised decision regression tree. The algorithm is based on an ensemble bagging technique, also known as bootstrap aggregation. Bagging is a technique where multiple

subsets are created for the training by sampling with replacement. Each subset is used to train a separate decision tree. The Bagging technique helps to reduce overfitting and leads to more robust and generalized predictions. Therefore, the RF algorithm creates multiple random decision trees during the training, each trained on a random subset of data. The final prediction $f(x)$ result to be the average of the predictions from all trees:

$$f(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \tag{5}$$

where B are the different bootstrapped training datasets, T_b are the regression tree predictors, and x are the unseen samples. One of the key advantages of the bagging technique used in Random Forest is its ability to reduce variance. This can be expressed mathematically as $Var(f_{RF}(x)) = \frac{1}{B}Var(T(x))$,where $Var(f_{RF}(x))$ is the variance of the overall Random Forest prediction, B is the number of trees in the forest and $Var(T(x))$ is the variance of a single decision tree predictor. This equation illustrates that increasing the number of trees B helps to reduce the overall variance of the model.

The main hyperparameters for random forest are the number of trees, the maximum number of features, the minimum number of samples required to split a node in a decision tree, the minimum number of samples required to be in a leaf node, the maximum depth of the decision trees in the forest, and the random number generator seed that is used to initialize the random forest algorithm.

XGBoost algorithm

The eXtreme Gradient Boosting is a supervised optimized boosting algorithm based on a gradient boosting minimization technique, similar to a neural network. It uses a regularization term which makes it efficient for most different types of datasets. In particular, the objective function $L(\phi)$ which needs to be optimized is given by the sum of the loss function and regularization term [46]:

$$L(\phi) = \sum_i l(\tilde{y}_i, y_i) + \sum_k \beta(f_k) \tag{6}$$

where l is the loss function representing the difference between the actual target value $\tilde{y}_i$ and the predicted target value y_i; $\beta(f_k)$ is the regularization term that is given by:

$$\beta(f_k) = \gamma T + 0.5 \cdot \lambda \cdot \omega^2 \tag{7}$$

where γ is the pruning index, T is the number of trees, λ is the scaling factor of the weights and ω is the leaf weight.

The algorithm tries to minimize l and optimizing regularization f_k leads to a smaller variance and makes the prediction stable. The algorithm during the training generates decision trees in series in a manner that the residuals of the previous tree are used for the training of the following one. The efficiency of the XGBoost algorithm is influenced by a lot of hyperparameters and the wrong choice of these can lead to poor performance. Key hyperparameters include: objective, learning_rate, n_estimators, booster, `max_depth`, `min_child_weight`, `gamma`, `subsample`, `colsample_bytree`, `reg_alpha (L1 Regularization)` and `reg_lambda (L2 Regularization)`. It is decided to use the default hyperparameters also because in this case find the best combination is computationally expensive (Table 2).

Table 2 Key XGBoost hyperparameters and default setting

Hyperparameters	Default value
objective	'reg:squarederror'
learning rate	0.3
n_estimators	100
subsample	1
booster	'gbtree'
max_depth	6
colsample_bytree	1
gamma	0
min_child_weight	1
reg_alpha (L1 Regularization)	0
reg_lambda (L2 Regularization)	1

Super Vector Regression algorithm

Support Vector Regression (SVR) is a regression algorithm based on Support Vector Machines (SVM) that finds the optimal hyperplane in a high-dimensional space with the maximum margin such that the data points (or as many as possible) lie within the margin and on the correct side of the hyperplane. The regression equation in the feature space is:

$$f(x) = < w, x > + bias \tag{8}$$

where $f(x)$ is the predicted output, x is the input feature vector, w is the weight vector, and $< \cdot, \cdot >$ denotes the dot product.

SVR can find nonlinear relationships by mapping the input features into a higher-dimensional space using a kernel function. It is useful for problems where the relationship between the independent variables and the dependent variable is nonlinear or complex. SVR aims to minimize the empirical risk while ensuring errors for each data point are within a tolerance ϵ, formulated as:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{9}$$

subject to the constraints:

$$\begin{aligned} y_i - \langle w, x_i\rangle - b &\le \epsilon + \xi_i \\ \langle w, x_i\rangle + b - y_i &\le \epsilon + \xi_i^* \\ \xi_i, \xi_i^* &\ge 0 \end{aligned} \tag{10}$$

where C is the regularization parameter, ξ_i and ξ_i^* are slack variables representing deviation from the true value. The parameter ϵ controls the width of the margin. It allows some tolerance for errors, known as ϵ-insensitive loss.

Data points within the margin or on the correct side of the hyperplane but outside the margin are considered to have no error. The regularization parameters allow the balance of the trade-off between maximizing the margin and minimizing the error. This helps prevent overfitting and ensures better generalization to unseen data.

Kernel Ridge Regression algorithm

The structure of the KRR algorithm is identical to the SVR [52]. The substantial difference consists in the adopted loss function. The SVR algorithm uses ϵ-insensitive loss combined with the L2 regularization (or ridge regularization) method [53]. KRR combines ridge regression with the Kernel trick to build a decision surface to map all data points randomly distributed in a multidimensional space. The Kernel trick is a method that allows finding a surface with the maximum margin that separates the data (not linearly separable), converting the problem into a higher-dimensional feature space where the data may be linearly separated. Mathematically, the Kernel trick allows to operate in the multidimensional feature space without calculating the coordinates. When an algorithm based on the Kernel trick is used, it is important to choose the adequate Kernel function $K(X_1, X_2)$ that maps the input data vectors in the higher dimensional space [54]. There are several Kernel functions to do this: linear, polynomial, radial basis, Gaussian, Laplacian, exponential chi2, and sigmoid. The Kernel function that best performs to take low-dimensional input space and transform it into a higher-dimensional space for our problem is the radial basis function (RBF):

$$K(X_1, X_2) = e^{-\gamma||X_1 - X_2||^2} \tag{11}$$

where X_1 and X_2 are input data points and $||X_1 - X_2||^2$ is the squared Euclidean distance. γ is a hyperparameter that is defined as $\frac{1}{2\sigma^2}$, where σ is a parameter known as the bandwidth of the kernel function. In this case, we set RBF as the kernel function whereas other hyperparameters are in the default set.

Another point is that the KRR technique generally slips into overfitting, but this problem can be prevented by adopting the cross-validation technique. In addition, KRR became faster than SVR for medium-sized datasets, i.e., less than one thousand samples, whereas SVR is better for larger training datasets. Considering the training time SVR is faster for all sizes of datasets because the learned solution is sparse [53].

k-Nearest Neighbors regression

k-nearest neighbors (kNN) regression is a non-parametric supervised learning method for prediction problems. The kNN regressor uses the proximity of data points in the feature space for the prediction. In particular, the kNN algorithm calculates its distance (e.g., by Euclidean distance, Manhattan distance, Minkowski distance, Hamming distance, etc.) from the "k" nearest data points to determine which group of data points it belongs to [53]. The default metric Minkowski was used for the distance computation:

$$d(X, Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}} \tag{12}$$

where $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$ are data points with n features and p is the power parameter for the Minkowski metric equal to 2 (default value). The kNN algorithm is based on the principle that similar data points tend to have similar target values. By finding the k nearest neighbors to a new data point, it makes predictions considering the average of these neighbors' target values.

Artificial Neural Network

The structure of an ANN is similar to a human brain. This is generally composed of an input layer of artificial neurons, one or more hidden layers, and an output layer. The input layer receives the information from the environment, but it is not processed. The information from the input layer is sent to the first hidden layer. Each neuron of a hidden layer receives inputs from all neurons of the previous layer. Each input information p_i is multiplied with a weight w_i, adds a *bias* term, and then passes the result through an activation function. The neural response can be mathematically expressed as:

$$n = \sum_{i=1}^{k} p_i w_i + bias \tag{13}$$

The *bias* is considered a systematic error that occurs in the ANN algorithm due to incorrect assumptions that occur during the training process. Technically, bias is defined as the error between the average algorithm prediction and the

ground truth. The activation function has an important role in the learning capability of ANN because it allows us to find non-linear relationships between the input and output. Common activation functions are the sigmoid, tanh, and ReLU (Rectified Linear Unit). Usually, the ReLU is the most used function because is able to converge faster compared to traditional activation functions since the derivative of ReLU is 1 for a positive input. The output of the ReLU function is the maximum value between 0 and the input value. The output layer receives the information from the previously last hidden layer and provides the output answer. The training is based on a backpropagation process that involves adjusting the weights and biases of the neurons. During the backpropagation process, the gradient of the loss function with respect to each weight and bias is calculated and then adjusted using Gradient Descent. The gradient descent is an optimization technique and the most popular are the stochastic gradient descent (SGD), the root mean squared propagation (RMSProp), and the adaptive moment estimation (Adam). The latter was used to minimize the loss function of our ANN. In this study, the ANN was constructed and trained using the framework PyTorch. The structure (i.e., number of neurons and layers) of an ANN is often found through a *trial and error* procedure. Generally, is important to have a large enough network for good learning but small enough to make a fast learning. A common issue for this type of algorithm is overfitting which represents the learning case where the performance of training is better than validation. This can be prevented by adopting techniques such as regularization methods (i.e., L1 or L2), dropout layer, and early stopping that help the algorithm to generalize better. The structure of our ANN was built using the Sequential module, which creates a stack of layers where each layer has exactly one input tensor and one output tensor. The ANN was built using the trial and error procedure, starting from a network made with a few resources (i.e., a few neurons and hidden layers) that are progressively increased until the test error does not improve anymore. This procedure allowed us to achieve the best one in performance and fast learning. The network consists of three hidden layers and one Dropout layer. The Dropout layer turns off the 30% of neurons helping to avoid overfitting (Table 3).

The hyperparameters as the learning rate are set before the training. The value of the learning rate defines the speed with which the network updates its parameters. For instance, a larger batch size might require a smaller learning rate to prevent overshooting of the optimal point, whereas a smaller batch size might require a larger learning rate to escape local minima. In our case, the best learning rate was set equal to 0.0001. The number of training epochs was 2500. An epoch is a hyperparameter that represents how many times a learning algorithm iterates over the entire training dataset during the training. At the end of each training of a single batch of samples, the algorithm learning is reset and the training starts again.

Table 3 Sequenze of hidden layers of the ANN

Layer	Neurons
Hidden layer 1	450
Hidden layer 2	400
Dropout layer	–
Hidden layer 4	400

Acknowledgements L. G. Di Maggio, C. Gastaldi, D. A. Renzo, and C. Delprete wish to honor the memory of our dear coauthor, Franco Furgiuele, who passed on August 29, 2024, acknowledging his contagious passion for research, his ability to identify the essential, and his warmth-qualities that made him a great professor, researcher, and supervisor. This research has been supported by ICSC - Italian Research Center on High Performance Computing, Big Data and Quantum Computing, funded by the European Union - NextGenerationEU.

Author contributions Luigi Gianpio Di Maggio: Conceptualization, Methodology, Software, Writing-original draft. Chiara Gastaldi: Conceptualization, Methodology, Supervision, Project Administration, Writing-original draft, Writing-review & editing. Danilo Antonello Renzo: Conceptualization, Methodology, Software, Writing-original draft. Cristiana Delprete: Conceptualization, Methodology, Writing-review & editing, Supervision. Franco Furgiuele: Conceptualization, Methodology, Supervision, Writing-review & editing.

Funding Open access funding provided by Politecnico di Torino within the CRUI-CARE Agreement.

Data availability Data will be made available upon request.

Declarations

Conflict of interest No potential conflict of interest is reported by the authors when submitting the work.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Gibson I, Rosen D, Stucker B, Khorasani M (2021) Additive manufacturing technologies. Springer, Cham. https://doi.org/10.1007/978-3-030-56127-7
2. Qin J, Hu F, Liu Y, Witherell P, Wang CCL, Rosen DW, Simpson TW, Lu Y, Tang Q (2022) Research and application of machine

learning for additive manufacturing. Addit Manuf 52:102691. https://doi.org/10.1016/j.addma.2022.102691
3. Esmaeilizadeh R, Keshavarzkermani A, Ali U, Behravesh B, Bonakdar A, Jahed H, Toyserkani E (2021) On the effect of laser powder-bed fusion process parameters on quasi-static and fatigue behaviour of Hastelloy X: a microstructure/defect interaction study. Addit Manuf 38:101805. https://doi.org/10.1016/j.addma.2020.101805
4. Yadollahi A, Mahmoudi M, Elwany A, Doude H, Bian L, Newman JC Jr (2020) Fatigue-life prediction of additively manufactured material: effects of heat treatment and build orientation. Fatigue Fract Eng Mater Struct 43(4):831–844
5. Zhan Z, Li H (2021) Machine learning based fatigue life prediction with effects of additive manufacturing process parameters for printed ss 316l. Int J Fatigue 142:105941. https://doi.org/10.1016/j.ijfatigue.2020.105941
6. Maleki E, Bagherifard S, Razavi N, Bandini M, du Plessis A, Berto F, Guagliano M (2022) On the efficiency of machine learning for fatigue assessment of post-processed additively manufactured alsi10mg. Int J Fatigue 160:106841. https://doi.org/10.1016/j.ijfatigue.2022.106841
7. Zhou T, Song Z, Sundmacher K (2019) Big data creates new opportunities for materials research: a review on methods and applications of machine learning for materials design. Engineering 5(6):1017–1026. https://doi.org/10.1016/j.eng.2019.02.011
8. Zhang M, Sun C-N, Zhang X, Goh PC, Wei J, Hardacre D, Li H (2019) High cycle fatigue life prediction of laser additive manufactured stainless steel: a machine learning approach. Int J Fatigue 128:105194. https://doi.org/10.1016/j.ijfatigue.2019.105194
9. Luo YW, Zhang B, Feng X, Song ZM, Qi XB, Li CP, Chen GF, Zhang GP (2021) Pore-affected fatigue life scattering and prediction of additively manufactured Inconel 718: an investigation based on miniature specimen testing and machine learning approach. Mater Sci Eng A 802:140693. https://doi.org/10.1016/j.msea.2020.140693
10. He L, Wang Z, Akebono H, Sugeta A (2021) Machine learning-based predictions of fatigue life and fatigue limit for steels. J Mater Sci Technol 90:9–19. https://doi.org/10.1016/j.jmst.2021.02.021
11. Bao H, Wu S, Wu Z, Kang G, Peng X, Withers PJ (2021) A machine-learning fatigue life prediction approach of additively manufactured metals. Eng Fract Mech 242:107508. https://doi.org/10.1016/j.engfracmech.2020.107508
12. Balamurugan R, Chen J, Meng C, Liu Y (2024) Data-driven approaches for fatigue prediction of Ti-6Al-4V parts fabricated by laser powder bed fusion. Int J Fatigue. https://doi.org/10.1016/j.ijfatigue.2024.108167
13. Gao J, Heng F, Yuan Y, Liu Y (2023) A novel machine learning method for multiaxial fatigue life prediction: improved adaptive neuro-fuzzy inference system. Int J Fatigue. https://doi.org/10.1016/j.ijfatigue.2023.108007
14. Zhu S, Zhang Y, Zhu B, Zhang J, He Y, Xu W (2024) High cycle fatigue life prediction of titanium alloys based on a novel deep learning approach. Int J Fatigue 182:108206. https://doi.org/10.1016/j.ijfatigue.2024.108206
15. Han S, Dinh TD, De Baere I, Boone M, Josipovic I, Van Paepegem W (2023) Study of the effect of defects on fatigue life prediction of additive manufactured Ti-6Al-4V by combined use of micro-computed tomography and fracture-mechanics-based simulation. Int J Fatigue. https://doi.org/10.1016/j.ijfatigue.2023.108110
16. Markham MJ, Fatemi A, Phan N (2024) Mixed-mode small fatigue crack growth rates and modeling in additively manufactured metals. Int J Fatigue. https://doi.org/10.1016/j.ijfatigue.2024.108258
17. Li H, Sun G, Tian Z, Huang K, Zhao Z (2024) A physics-informed neural network framework based on fatigue indicator parameters for very high cycle fatigue life prediction of an additively manufactured titanium alloy. Fatigue Fract Eng Mater Struct 47(9):3171–3188. https://doi.org/10.1111/ffe.14363
18. Zhang Z, Xu Z (2023) Fatigue database of additively manufactured alloys. Sci Data. https://doi.org/10.1038/s41597-023-02150-x
19. Wang H, Li B, Gong J, Xuan F-Z (2023) Machine learning-based fatigue life prediction of metal materials: perspectives of physics-informed and data-driven hybrid methods. Eng Fract Mech 284:109242. https://doi.org/10.1016/j.engfracmech.2023.109242
20. Lian Z, Li M, Lu W (2022) Fatigue life prediction of aluminum alloy via knowledge-based machine learning. Int J Fatigue 157:106716. https://doi.org/10.1016/j.ijfatigue.2021.106716
21. Jäger S, Allhorn A, Bießmann F (2021) A benchmark for data imputation methods. Front Big Data 4:693674
22. Jadhav A, Pramod D, Ramanathan K (2019) Comparison of performance of data imputation methods for numeric dataset. Appl Artif Intell 33(10):913–933
23. Ciampaglia A, Tridello A, Paolino DS, Berto F (2023) Data driven method for predicting the effect of process parameters on the fatigue response of additive manufactured AlSi10Mg parts. Int J Fatigue 170:107500. https://doi.org/10.1016/j.ijfatigue.2023.107500
24. Lai W-J, Ojha A, Li Z (2022) Effect of post heat treatment on fatigue strength of AlSi10mg produced by laser powder bed fusion process. In: TMS 2022 151st annual meeting & exhibition supplemental proceedings. Springer, Cham, pp 141–163
25. Qin Z, Kang N, El Mansori M, Wang Z, Wang H, Lin X, Chen J, Huang W (2022) Anisotropic high cycle fatigue property of Sc and Zr-modified Al–Mg alloy fabricated by laser powder bed fusion. Addit Manuf 49:102514. https://doi.org/10.1016/j.addma.2021.102514
26. Damon J, Hanemann T, Dietrich S, Graf G, Lang K-H, Schulze V (2019) Orientation dependent fatigue performance and mechanisms of selective laser melted maraging steel X3NiCo-MoTi18-9-5. Int J Fatigue 127:395–402. https://doi.org/10.1016/j.ijfatigue.2019.06.025
27. Witkin DB, Patel D, Albright TV, Bean GE, McLouth T (2020) Influence of surface conditions and specimen orientation on high cycle fatigue properties of Inconel 718 prepared by laser powder bed fusion. Int J Fatigue 132:105392. https://doi.org/10.1016/j.ijfatigue.2019.105392
28. Kotzem D, Kleszczynski S, Stern F, Elspaß A, Tenkamp J, Witt G, Walther F (2021) Impact of single structural voids on fatigue properties of AISI 316L manufactured by laser powder bed fusion. Int J Fatigue 148:106207. https://doi.org/10.1016/j.ijfatigue.2021.106207
29. Mishurova T, Artzt K, Rehmer B, Haubrich J, Àvila L, Schoenstein F, Serrano-Munoz I, Requena G, Bruno G (2021) Separation of the impact of residual stress and microstructure on the fatigue performance of LPBF Ti-6Al-4V at elevated temperature. Int J Fatigue 148:106239. https://doi.org/10.1016/j.ijfatigue.2021.106239
30. Merot P, Morel F, Gallegos Mayorga L, Pessard E, Buttin P, Baffie T (2021) Observations on the influence of process and corrosion related defects on the fatigue strength of 316L stainless steel manufactured by laser powder bed fusion (L-PBF). Int J Fatigue 155:106552. https://doi.org/10.1016/j.ijfatigue.2021.106552
31. Sausto F, Carrion PE, Shamsaei N, Beretta S (2022) Fatigue failure mechanisms for alsi10mg manufactured by L-PBF under axial and torsional loads: the role of defects and residual stresses. Int J Fatigue 162:106903. https://doi.org/10.1016/j.ijfatigue.2022.106903
32. Cacace S, Gökhan Demir A, Sala G, Mattia Grande A (2022) Influence of production batch related parameters on static and fatigue resistance of LPBF produced AlSi7Mg0.6. Int J Fatigue 165:107227. https://doi.org/10.1016/j.ijfatigue.2022.107227

33. Yan X, Yin S, Chen C, Huang C, Bolot R, Lupoi R, Kuang M, Ma W, Coddet C, Liao H, Liu M (2018) Effect of heat treatment on the phase transformation and mechanical properties of Ti6Al4V fabricated by selective laser melting. J Alloy Compd 764:1056–1071. https://doi.org/10.1016/j.jallcom.2018.06.076
34. Kong W, Francis EM, Shi Q, Cox SC, Wang F, Kuang M, Attallah MM (2022) The influence of advanced hot isostatic pressing on phase transformations, mechanical properties of Ti-34Nb-13Ta-5Zr-0.2O alloy manufactured by in-situ alloying via selective laser melting. J Alloys Compd 903:163974. https://doi.org/10.1016/j.jallcom.2022.163974
35. Beretta S, Patriarca L, Gargourimotlagh M, Hardaker A, Brackett D, Salimian M, Gumpinger J, Ghidini T (2022) A benchmark activity on the fatigue life assessment of AlSi10Mg components manufactured by L-PBF. Mater Design 218:110713. https://doi.org/10.1016/j.matdes.2022.110713
36. Xu ZW, Wang Q, Wang XS, Tan CH, Guo MH, Gao PB (2020) High cycle fatigue performance of alsi10mg alloy produced by selective laser melting. Mech Mater 148:103499. https://doi.org/10.1016/j.mechmat.2020.103499
37. Han Q, Mertens R, Montero-Sistiaga ML, Yang S, Setchi R, Vanmeensel K, Van Hooreweder B, Evans SL, Fan H (2018) Laser powder bed fusion of Hastelloy X: effects of hot isostatic pressing and the hot cracking mechanism. Mater Sci Eng A 732:228–239. https://doi.org/10.1016/j.msea.2018.07.008
38. Zhang C, Liu F, Wang H (2022) Effect of the prior-β grain boundaries on mechanical behavior of Ti6Al4V alloy processed by laser directed energy deposition. Mater Sci Eng A 848:143389. https://doi.org/10.1016/j.msea.2022.143389
39. Brika SE, Brailovski V (2020) Influence of powder particle morphology on the static and fatigue properties of laser powder bed-fused Ti-6Al-4V components. J Manuf Mater Proc. https://doi.org/10.3390/jmmp4040107
40. Jimenez EH, Kreitcberg A, Moquin E, Brailovski V (2022) Influence of post-processing conditions on the microstructure, static, and fatigue resistance of laser powder bed fused Ti-6Al-4V components. J Manuf Mater Proc. https://doi.org/10.3390/jmmp6040085
41. Voloskov B, Evlashin S, Dagesyan S, Abaimov S, Akhatov I, Sergeichev I (2020) Very high cycle fatigue behavior of additively manufactured 316L stainless steel. Materials. https://doi.org/10.3390/ma13153293
42. Klein Fiorentin F, Maciel D, Gil J, Figueiredo M, Berto F, Jesus A (2022) Fatigue assessment of Inconel 625 produced by directed energy deposition from miniaturized specimens. Metals. https://doi.org/10.3390/met12010156
43. Bassoli E, Denti L, Comin A, Sola A, Tognoli E (2018) Fatigue behavior of as-built L-PBF A357.0 parts. Metals. https://doi.org/10.3390/met8080634
44. Zhan Z, Li H (2021) A novel approach based on the elastoplastic fatigue damage and machine learning models for life prediction of aerospace alloy parts fabricated by additive manufacturing. Int J Fatigue 145:106089. https://doi.org/10.1016/j.ijfatigue.2020.106089
45. Horňas J, Běhal J, Homola P, Senck S, Holzleitner M, Godja N, Pàsztor Z, Hegedüs B, Doubrava R, Růžek R, Petrusovà L (2023) Modelling fatigue life prediction of additively manufactured Ti-6Al-4V samples using machine learning approach. Int J Fatigue 169:107483. https://doi.org/10.1016/j.ijfatigue.2022.107483
46. Konda N, Verma R, Jayaganthan R (2023) Estimation of high cycle fatigue life of additively manufactured Ti6Al4V using data analytics. Procedia Struct Integr 46:87–93
47. Géron A (2018) Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems, first edition, fifth release edition. O'Reilly, Beijing Boston Farnham Sebastopol Tokyo
48. Horňas J, Běhal J, Homola P, Senck S, Holzleitner M, Godja N, Pàsztor Z, Hegedüs B, Doubrava R, Růžek R, Petrusovà L (2023) Modelling fatigue life prediction of additively manufactured Ti-6Al-4V samples using machine learning approach. Int J Fatigue 169:107483. https://doi.org/10.1016/j.ijfatigue.2022.107483
49. He L, Wang Z, Akebono H, Sugeta A (2021) Machine learning-based predictions of fatigue life and fatigue limit for steels. J Mater Sci Technol 90:9–19. https://doi.org/10.1016/j.jmst.2021.02.021
50. Maleki E, Bagherifard S, Razavi N, Bandini M, Plessis AD, Berto F, Guagliano M (2022) On the efficiency of machine learning for fatigue assessment of post-processed additively manufactured AlSi10Mg. Int J Fatigue 160:106841. https://doi.org/10.1016/j.ijfatigue.2022.106841
51. Xu Z, Zhang Z (2024) The need for standardizing fatigue data reporting. Nat Mater 23(7):866–868. https://doi.org/10.1038/s41563-024-01929-6
52. Vovk V (2013). In: Schölkopf B, Luo Z, Vovk V (eds) Kernel ridge regression. Springer, Berlin, Heidelberg, pp 105–116
53. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
54. Stulp F, Sigaud O (2015) Many regression algorithms, one unified model: a review. Neural Netw 69:60–79. https://doi.org/10.1016/j.neunet.2015.05.005

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.