Machine Learning of Engine Health Monitoring Data: Development of a machine learning model for damage prediction of real flight missions

TECHNISCHE UNIVERSITÄT BERLIN

Faklutät V für Verkehrs- und Maschinensysteme
im Fach Festigkeit und Lebensdauer

Bachelorarbeit

zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.)
im Studiengang Physikalische Ingenieurwissenschaft

Verfasser	Samuel Oliver Bryson
Matrikelnr.	352667
Ausgabedatum	29.01.2020
Eingereicht am	XX.XX.2020
Betreuung	Prof. Dr.-Ing. Robert Liebich Dr. Stephan Pannier Tolga Yağcı, M.Eng. Tobias Werder, M.Sc.

Eidesstattliche Erklärung

Hiermit erkläre ich, Samuel Oliver Bryson (geboren am 21.12.1991 in Nottingham, Großbritannien), dass ich die vorliegende Abschlussarbeit eigenständig und ohne unerlaubte fremde Hilfe angefertigt habe sowie keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Berlin, den XX.XX.XXX ______________________________

Samuel Oliver Bryson

Contents

1 Introduction 6

2 Theoretical Background 8

2.1 Definitions 8

2.2 Big Data 9

2.3 Machine Learning 9

2.3.1 Polynomial Regression 10

2.4 Deep Learning 11

2.4.1 Neural Networks 12

2.4.2 Time Series Classification 15

2.4.3 Training and Validating a Neural Network 16

2.4.4 Overfitting and Underfitting 17

2.5 The Engine 18

2.6 Damage 20

2.6.1 Certification 21

2.6.2 Exchange Rate 22

2.7 Research Question 22

3 Data Analysis 24

3.1 Engine Health Monitoring Data 24

3.2 Flight Phases 26

3.3 Features 28

3.4 Cycle Counter 29

3.5 Downsampling 29

3.5.1 (A)PLA 29

3.5.2 APCA 29

3.5.3 Data Loss 29

3.6 Dataset Reduction 32

4 Model Implementation **33**
4.1 Polynomial Regression on Key Values 33
4.2 MLP: Key Value Regression 34
4.3 MLP: Time Series Regression 34
4.4 Convolutional Neural Network: Time Series Classification 34
    4.4.1 Four classes 35
    4.4.2 Four classes with gap 35
    4.4.3 Ten classes 35
4.5 Convolutional Neural Network: Time Series Regression 35
    4.5.1 Time Series Regression 36

5 Discussion **37**

6 Conclusion **38**

List of Abbreviations

AI artificial intelligence

CNN convolutional neural network

CSV comma-separated values

DL Deep learning

EASA European Union Aviation Safety Agency

EHM Engine Health Monitoring

FAA Federal Aviation Authority

FE Finite Element

FM flight mode

HP high-pressure

HPT high-pressure turbine

IP intermediate-pressure

LP low-pressure

LTFC Life To First Crack

ML machine learning

MLP multilayer perceptron

MTS multivariate time series

PSCL Predicted Safe Cyclic Life

ReLU Rectified Linear Unit

TET turbine entry temperature

TRA thrust lever resolver angle

TSC Time series classification

WOW weight on wheels

1 Introduction

In recent years, many industries have experienced enormous changes due to an influx of huge quantities of data, increasingly affordable data storage solutions and access to high computational power (Chen et al., 2014). The aviation industry is no exception: the Airbus A350 XWB comes equipped with approximately 6 000 sensors that produce 300 GB of data every day; the next generation Airbus A380 will come with 10 000 sensors on a single wing alone (Rajaraman, 2016).

This data can only be of value if there are suitable tools for handling it. The arrival of the age of Big Data (Fan & Bifet, 2013) coincided unsurprisingly with an increase in the popularity of artificial intelligence (AI) and machine learning (ML). Today, highly optimised, specially designed programming libraries make data-orientated ML approaches more accessible and more powerful than ever.

Operators using aircraft powered by engines made by Rolls-Royce can return Engine Health Monitoring (EHM) data to the company on a voluntary basis. This data is recorded during flights and includes parameters such as temperature and pressure at various stages of the engine, flight altitude and speed, and many others.

Rolls-Royce uses this data for various analyses, such as determining the amount of service life consumed during the flight mission. Currently, the process involves hand-selecting individual points of interest (features) of a component, and performing a semi-automated fatigue analysis on each of these individually, from which the consumed service life is determined. This method has been sufficient in the past due to its robustness and efficiency for a small number of features, but is limited by the difficulties and time involved in selecting features a priori.

This thesis, written in cooperation with Rolls-Royce Deutschland, aims to identify, evaluate and compare a number of ML-based methods that avoid this manual input stage by producing an output for *all* component surface nodes with comparable accuracy and speed to the current method.

The thesis is structured as follows: First, the theoretical background will be covered alongside an introduction to subfields of ML and Deep learning (DL) and state-of-the-art methods of interest for the research. Then, an overview of the input data will be presented. After this, the methods identified will be implemented and their results presented in the practical section, with a subsequent evaluation and comparison in the discussion. Finally, in the conclusion, suggestions will be made for further research into the topic.

2 Theoretical Background

2.1 Definitions

A **univariate time series** X of length l is a sequential collection of values x_i given by

$$X = [x_1, x_2, \dots, x_l]$$

where $x_i \in \mathbb{R}$, $i \in \mathbb{N}$ with $i \in [1, l]$.

An **m-dimensional multivariate time series (MTS)** X of length l is a collection of m time series of length l. (Ismail Fawaz et al., 2019)

In the practical part of this thesis, a **dataset** D of length n is a set of n pairs of input data X_i and corresponding output data Y_i, i.e.:

$$D = \left\{ \left(X_1,\, Y_1\right), \left(X_2,\, Y_2\right), \dots, \left(X_n,\, Y_n\right) \right\}$$

where for all $i \in [1, n]$, X_i is a univariate or multivariate time series, and Y_i contains the given output values corresponding to X_i.

A **sample** is a single pair $(X_i,\, Y_i)$ for some $i \in [1, l]$, and i is referred to as its **index**.

In a classification task, for some sample with index i, Y_i will have length k where k is the number of possible classes that could be attributed to X_i. Additionally, all $Y_{i,j} = 0$ for each class index $j \in [1, k]$ except where j is given as the class of X_i, in which case $Y_{i,j} = 1$.

In a regression task, Y_i has length k where k is the number of output values desired and each $Y_{i,j} \in \mathbb{R}$ for $j \in [1, k]$.

2.2 Big Data

The term *Big Data* describes enormous datasets have become the norm over the past two decades due to the ubiquity of data-collecting devices and their increasing connectivity. It covers datasets of such great size that new methods are required to organise and extract useful information from them, as the data is often unstructured and noisy (Fan & Bifet, 2013; Chen et al., 2014).

One increasingly important, as well as highly challenging (Yang & Wu, 2006), data type is that of time series. A time series is a set of values ordered chronologically. It is often used in analyses for detecting trends and making predictions (on financial data, for example (Krollner et al., 2010)), but, within a ML context, has in recent years been used for classification (Dau et al., 2019; Ismail Fawaz et al., 2019) and, in this thesis, regression.

2.3 Machine Learning

The desire to extract value from such datasets has led to an enormous increase in the popularity of machine learning (ML).

ML is a subfield of AI, the field of research that occupies itself with giving machines the ability to think and learn. In ML, machines attempt to extract information and patterns from data without thorough or explicit instructions, often making use of highly contrived data. Classifying objects based on a finite set of selected input values (e.g. birds based on their weight, wingspan, the colour of their back and whether they have webbed feet (Harrington, 2012)) is a task for which machine learning is regularly used.

ML tasks are generally split into two categories: supervised and unsupervised learning. The former involves training a model to associate input data with known output data in order to use the model on previously unseen data; in the latter, a model is given data and instructed to find patterns or groups based on input data alone (Goodfellow et al., 2016; Kelleher et al., 2015).

2.3.1 Polynomial Regression

Although regression analysis has its roots in statistics, it has great overlaps with ML due to its root objective: to map a number of inputs (the independent variables x_1, x_2, …, x_n) to a given continuous output (the dependent variables y_1, y_2, …, y_n), and to predict outputs for previously unseen input data. Since the dependent variable data is known in the training data, regression analysis is a type of supervised machine learning.

A (n-dimensional) linear regression maps a linear model

$$y = a_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n$$

to given data and attempts to find the parameters a_i, $i \in [0, n]$ that provide the best fit. In the Python library `scikit-learn`, the `LinearRegression` class[1] determines these by minimising the residual sum of squares between the given samples and the values predicted by the fitted model.

For non-linear relationships, each x_i can be transformed using some basis function g_i after which the coefficients a_i can be determined in the same way as in the linear case. A polynomial regression is carried out by defining the basis function as $g(x_i) = x^i$, leading to the polynomial function

$$f(x) = y = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n.$$

It should be noted that, although the function is now polynomial, the model is still linear since the basis function only serves to transform the input data to a higher dimension; after the transformation using `scikit-learn`'s `PolynomialFeatures` class, the `LinearRegression` class will determine the coefficients a_1, …, a_n as before.

---

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`

The performance of a regression is usually measured using the coefficient of determination, R^2, which compares the predicted output values to the mean of the expected output data $\bar{y}$, and is given by

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

where

$$SS_{\text{res}} = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

is the residual sum of squares, and

$$SS_{\text{tot}} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

is the total sum of squares. According to this measure, a perfect fit is indicated by a performance of $R^2 = 1$, a fit that is no better than the mean of the output data by $R^2 = 0$, and a fit that is even worse than the mean by $R^2 < 0$.

2.4 Deep Learning

DL is an application of ML that employs more complex models capable of making sense of noisier, largely unprocessed data, such as sound signals and images (Goodfellow et al., 2016). A bird classification problem could in this case involve training a model to extract necessary information from an image of a bird and associating this with the bird's label, requiring only the correct labelling of the image with no manual measurements.

The great advantage of DL over ML is its ability to make sense of complex data using simpler representations in a hierarchical system (Goodfellow et al., 2016). Returning to the example of bird classification: To extract information on a bird from a photo, a trained model might analyse an input image for contours and corners; in the next layer, these could be combined to recognise edges, which a subsequent layer could recognise as beaks or wings, and so on. To reach this stage, however, the architecture of the model must be suitable and the model itself must be trained using enough input data of sufficient quality. To achieve an acceptable level of accuracy, models should be trained on roughly 5 000 samples per class, with several orders of magnitude more required to achieve human-level accuracy (Goodfellow et al., 2016, p. 20).

DL has seen a huge rise in popularity in recent years, with applications including speech recognition (Deng & Li, 2013), medical diagnoses (Lee et al., 2018), stock market prediction (Krollner et al., 2010) and many others (Kelleher et al., 2015). One particular challenge among the DL research community is time series data (Yang & Wu, 2006), a sequential collection of values recorded over time. Time series data remains a great challenge due to its noisy, multidimensional nature (Kelleher et al., 2015) and the dfficulties involved in developing algorithms that can also interpret the temporal information it holds (Bagnall et al., 2017).

2.4.1 Neural Networks

Two types of neural networks are relevant to this thesis: the multilayer perceptron (MLP) and the convolutional neural network (CNN). The fundamental element of the both of these is the perceptron.

2.4.1.1 Perceptron

The perceptron is a simple mathematical function somewhat inspired by biological neurons Rosenblatt (1958). A perceptron (Figure 1) takes the sum of n values x_1, x_2, …, x_n multiplied with their respective weights w_1, w_2, …, w_n as its input. This input value is added to the perceptron's bias b and fed into its activation function f. The result of this function is the perceptron's output value z which is fed into the next layer of the network. This is described by the following formula:

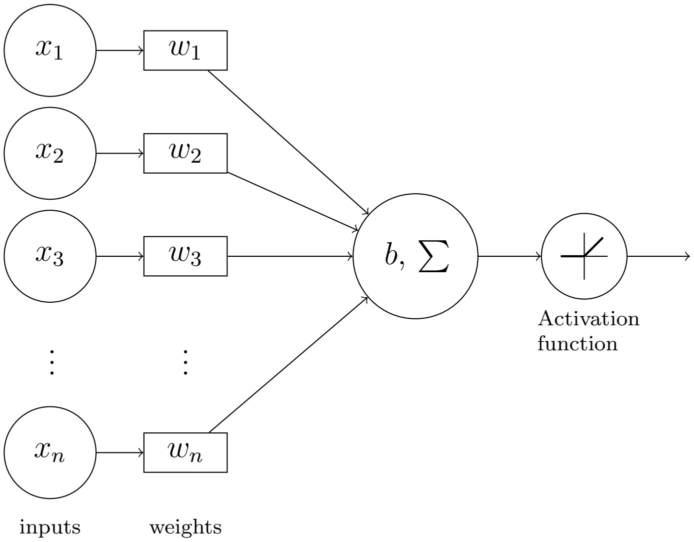

Figure 1: A perceptron takes the sum of the values x_i multiplied with their respective weights w_i, adds to this a bias b and plugs this into an activation function f; the output value is sent to the next layer of the MLP.

$$z = f\left(b + \sum_{i=1}^{n} x_i w_i\right)$$

The activation function f is set during the definition of the network and can take many forms, such as the identity ($f(x) = x$), sigmoid ($f(x) = 1/\left(1 + e^{-x}\right)$) and Rectified Linear Unit (ReLU) ($f(x) = \max(0, x)$) (Hahnloser et al., 2000). ReLU has been found to produce the best results (Jarrett et al., 2009) and is used in all hidden layers of MLPs in the models built in this thesis.

2.4.1.2 Multilayer perceptron

The MLP, also known as a feedforward neural network or deep neural network, is the quintessential DL network. It comprises three main stages: an input layer, one or several hidden layers, and an output layer (see Figure 2).

The input layer is the entry point of the network for input data and consists of one

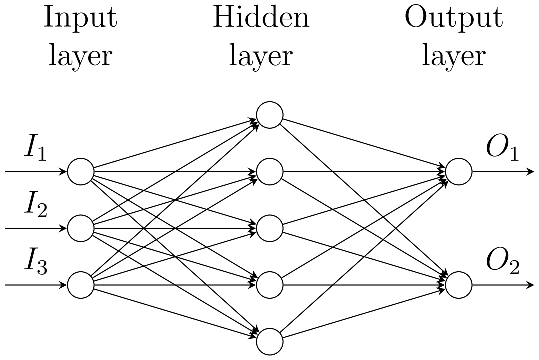

Figure 2: A multilayer perceptron with three nodes in the input layer, five fully-connected perceptrons in one hidden layer and three fully-connected output perceptrons in the output layer

node per input value. (The term *node* is used here to distinguish between its function and that of a perceptron: a node is purely a data entry point with no processing involved.) Generally, the hidden and output layers are fully connected, meaning that the input of each node in these layer is a combination of *all* of the outputs from the previous layer with their respective weights.

Input values X_{train} are entered into its input layer and fed forwards through each layer of the network, combined with all weights and biases of each perceptron and its activation function, until they reach the output layer.

The activation function of the output layer is determined by the task. In a classification task, the goal is for the MLP to predict the label most likely to apply to the input data from a finite number of labels, whereby the number of possible labels corresponds to the number of output nodes. For this, the softmax activation function is ideal (Goodfellow et al., 2016, p. 184) as it outputs a probability distribution, the highest value of which is taken to be the index of the predicted class.

In a regression task, each predicted output value should be a decimal value corresponding to the network's interpretation of the input data.

2.4.1.3 Convolutional neural network

A CNN is a neural network that makes use of convolutions and, in most models, pooling operations. Its qualities make it ideal for handling sequential, grid-like data such as temporal sensor data (one-dimensional grids of equally spaced readings) or images (two-dimensional grids of pixels). In particular, convolutions and pooling make it highly suitable for time-invariant (or space-invariant) pattern detection. (Goodfellow et al., 2016)

A convolution is a non-linear transformation of the sequential data that is used for feature detection. It can be thought of as sliding filter moving across the input data whose values are learned and optimised in a similar way to the weights and bias of perceptrons. However, while each element of an MLP has one weight value for each connection to the next layer, the convolution introduces the great advantage of *parameter sharing* as each filter value is used on all input data. Using convolutions means that a significantly lower number of parameters require training and storing, and improves efficiency by several orders of magnitude (Goodfellow et al., 2016).

Pooling is also a transformation used for statistical summaries of output values from the previous layer (Goodfellow et al., 2016). Pooling functions, such as max pooling (Zhou & Chellappa, 1988), bring the additional benefit of temporal (or, with images, spatial) invariance to slight translations of the input values, offering a flexibility that can be useful with noisy data.

One further type of neural network is a Recurrent Neural Network. While these are generally associated with time series data, they are used for forecasting future time series based on preceding data, which is not relevant to this thesis.

2.4.2 Time Series Classification

Time series classification (TSC) involves reading time series data and applying one or several labels to each instance Fawaz et al. (2019). The UCR Archive Dau et al. (2019) is a large collection of TSC datasets released to serve as a benchmark dataset for evaluating newly proposed DL approaches to TSC.

The greatest breakthroughs in TSC have only come about within the past few years since the publication of the Inception module (Szegedy et al., 2014) and its subsequent adaptation for application in further DL fields Ismail Fawaz et al. (2019); Fawaz et al. (2019).

It should be noted at this point that the various labels predicted in classification problems bear no relation to one another. Specifically, they are not ordinal: In an output layer with three output nodes (predicting the probability of five different labels), the first label is no more “similar” to the second than it is to the fifth.

2.4.3 Training and Validating a Neural Network

For supervised learning tasks, a dataset D of length n is split into two pairs: the training dataset $D_{\text{train}} = (X_{\text{train}}, Y_{\text{train}})$ of length n_{train}, and the validation dataset $D_{\text{val}} = (X_{\text{val}}, Y_{\text{val}})$ of length n_{val}, with $n_{\text{train}} + n_{\text{val}} = n$, whereby D_{train} and D_{val} are disjoint. (To avoid overcomplicating the study design, it was decided not to implement cross-validation in this thesis.)

Training a model involves iterating through all samples in D_{train} in batches of a defined batch size b. Each sample $X_{\text{train},i}$ is entered into the input layer and fed through the model to generate output values $Y_{\text{pred},i}$ as predicted by the parameters (weights, biases, filter values, etc.) within the model. These output values are then compared with the given output values $Y_{\text{train},i}$ by means of a loss function.

A loss function is used to quantify how well a model has performed. For a regression model, this often takes the form of mean squared error; for classification, generally some form of cross-entropy is applied. The goal of training is to minimise the output of the loss function, thereby minimising the difference between Y_{pred} and Y_{train}.

Using a training algorithm (such as backpropagation for an MLP), the model parameters are optimised to produce better output values in the next batch. This process is repeated for all training samples; one iteration through all training samples is called an epoch. (Kirk, 2017)

Training is carried out for a defined number of epochs. After each epoch, the model is tested on the validation dataset as an additional measure of performance: X_{val} is fed into the model to produce a new Y_{pred}, which is then compared with Y_{val} using the loss function. The crucial difference is that, during validation, the model is not optimised to ensure that the validation data is always “unseen”, meaning that it remains an unbiased performance measure throughout the entire training phase.

The lengths of the training and validation datasets, n_{train} and n_{val}, can be freely defined but, in general, increasing the number of training samples will lead to better model performance. However, if the number of validation samples n_{val} is too low, the reliability of the validation will be questionable. In the model implementation for this thesis (Section 4), models were trained on a training dataset of length $\frac{2}{3} \cdot n$ and validated using a validation dataset of length $\frac{1}{3} \cdot n$, rounding where necessary.

2.4.4 Overfitting and Underfitting

The more epochs for which a neural network is trained, the better it becomes at predicting the output on the training dataset. However, if the number of epochs is set too high and no perfect fit is achievable for the entire dataset (due to, for example, some element of randomness or noise), the model can become overly optimised for the training data to the detriment of its accuracy in predicting unseen data.

Similarly, in the context of a polynomial regression (Figure 3), overfitting can be caused by attempting to fit the model using an overly high-order polynomial, while underfitting is caused by using an insufficiently high-order polynomial. (VanderPlas, 2016, p. 365)

Success in ML tasks requires finding a balance between undercomplexity and overcomplexity in the model. To avoid overfitting (or underfitting) on neural networks, a suitable number of training epochs (among other hyperparameters) should be found.

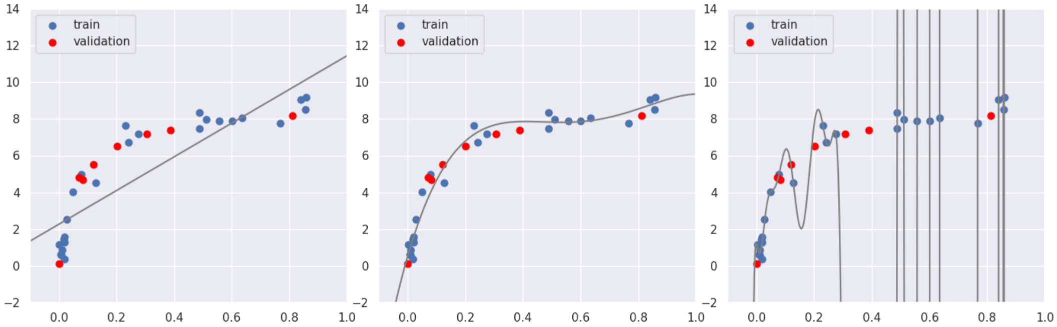

Figure 3: A polynomial regression fitted to random data with polynomials of order 1 (left, underfitting), 4 (centre, suitable) and 20 (right, overfitting). Training data is shown in blue, validation data in red. Overfitting leads to better predictions on training data, but poor predictions on validation data (based on (VanderPlas, 2016)).

2.5 The Engine

Companies active within civil aerospace companies design and manufactures high-bypass turbofan jet engines, which offer an ideal arrangement for civil aircraft flying below the speed of sound (plc, 2015).

Engines consist of four main components (fan, compressor, combustion chamber and turbine) which correspond approximately to the four stages of a Brayton cycle (intake, compression, combustion, expansion). The engine draws in air, compresses it and burns fuel in the compressed air to further increase the pressure. In this high-pressure, high-temperature state, the air expands and is blown out of the nozzle, driving the turbine on its way which in turn powers the fan and compressor (plc, 2015).

Many modern civil aircraft engines are equipped with two concentric shafts, split into low-pressure (LP) and high-pressure (HP), which connect respective sets of compressors and turbines (Spittle, 2003). Some, including the Rolls-Royce Trent family of engines, also have a third shaft, coupling the intermediate-pressure (IP) compressor and turbine. Figure 4 shows the configuration of such an engine.

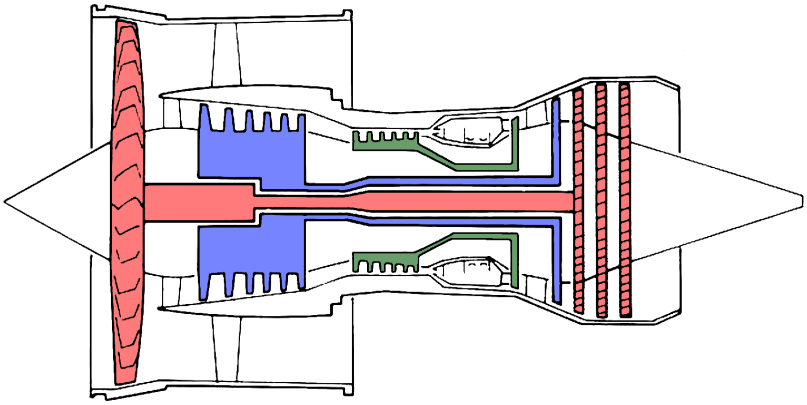

Figure 4: A schematic diagram of a triple-spool high-bypass engine with LP, IP and HP components shown in red, blue and green, respectively (based on (plc, 2015)).

A multi-spool configuration can greatly improve the efficiency of an engine, but comes with some drawbacks. The overall efficiency of an engine, $\eta_{overall}$, is the product of its thermal and propulsive efficiencies:

$$\eta_{overall} = \eta_{thermal} \cdot \eta_{propulsive}$$

Assuming an ideal Carnot cycle, the thermal efficiency is given by

$$\eta_{thermal} = 1 - \frac{T_{min}}{T_{max}}$$

whereby T_{min} is the engine entry temperature (assumed constant) and T_{max} the maximum temperature (reached within the combustion chamber). A higher T_{max} therefore leads to greater thermal efficiency. Increasing the temperature, however, requires higher compression. When the engine's overall pressure ratio (the ratio of stagnation pressure between the end and the beginning of the compressor stage) reaches values of 8 to 10, additional shafts become necessary (Bräunling, 2015, p. 58) in order to enable sufficiently high rotational speeds.

These requirements and developments have led to an average increase in turbine entry temperature (TET) of 10 K per year over the past 80 years (Kyprianidis, 2011), and in modern engines has reached temperatures far above the melting point of high-pressure turbine (HPT) blades, requiring great improvements in the materials, structures and systems used (Spittle, 2003). With this improvement in thermal efficiency, the arrangement comes with the compromise of significantly increased thermomechanical (due to higher temperature) and mechanical loads (due to higher rotational speeds) within the engine, particularly as the air passes from the combustion chamber to the HPT.

2.6 Damage

Components used in such extreme conditions experience degradation. They therefore cannot be used indefinitely and must be removed from service after a certain amount of time to avoid potentially catastrophic failure. The amount of time for which the component is permitted for service, referred to as its Approved Life, is determined in safety analyses (EASA, 2015). Approved Life is measured in Engine Flight Cycles, to be referred to as *cycles* in the following.

The degradation of the engine through its use is referred to as damage. The extent of damage is dependent on many parameters: Since operators use their aircraft for different purposes, flight parameters such as duration, altitude and outside temperature vary greatly and result in different levels of damage. This idea can be quantified with the aforementioned cycles.

In a Finite Element (FE) context, components are modelled digitally and separated into a finite number of individual elements defined by their corresponding corner points, or nodes. Areas of particular interest within the component (usually where stresses are expected to be highest) are referred to as features.

Within Rolls-Royce, damage from flight missions is currently calculated using one of two internal tools for flight profile analysis: SA66 and Perseus. The former is ideal for processing many flights in a short amount of time, but is restricted to a low number of features due to the time involved in manually setting up the surrogate FE model for each feature. The latter can be described as a brute-force method that determines damage for all surface nodes, but is currently restricted by the amount of time required to process a single flight.

2.6.1 Certification

Certification of a new engine includes a thorough safety analysis as described in EASA (2015).

Of primary concern is the HPT disk due to the extreme conditions under which it operates and the risk of Hazardous Engine Effects. The latter is defined to include (among others) 'non-containment of high-energy debris', 'uncontrolled fire' and 'complete inability to shut the engine down' according to the European Union Aviation Safety Agency (EASA) and the Federal Aviation Authority (FAA) EASA (2015); FAA (2007). A safety analysis must show that Hazardous Engine Effects are expected to occur with a probability no greater than 10^{-7} per flight hour.

Engine parts whose failure is likely to result in Hazardous Engine Effects are labelled Engine Critical Parts; the HPT disk also carries this label. Engine Critical Parts are assigned an Approved Life, which defines the 'mandatory replacement life' EASA (2015) of the part and is measured in Engine Flight Cycles, which are defined by a flight profile that is considered a reference flight mission, corresponding approximately to the average flight for which the engine is expected to be used in service.

Determining a part's Approved Life, commonly referred to as lifing, is a complex process, the majority of which involves 'defining the duty the part is required to sustain' Corran & Williams (2007), i.e. the design and refinement of the Engine Flight Cycle. One lifing philosophy is that of Life To First Crack (LTFC), which involves determining the Predicted Safe Cyclic Life (PSCL) by means of statistically-determined safety factors.

2.6.2 Exchange Rate

One value calculated from the EHM flight data and used as input data for regression models in this thesis is the exchange rate ER. This value is derived from the Palmgren-Miner rule (Palmgren, 1924) which states that a material will fail when the accumulated damage D, determined by the formula

$$D = \sum_i \frac{n_i}{N_i},$$

reaches the value of 1. Here, n_i is the number of cycles at a certain amplitude σ_i in a stress spectrum S, while N_i is the number of cycles to failure at σ_i as defined by the material's SN-curve.

The exchange rate is the proportion of D to the damage from the flight's major cycle, i.e. $n_{\max}/N_{\max}$ where the subscript $_{\max}$ denotes the index of the maximum stress amplitude in S. Therefore,

$$ER = \sum_i \frac{n_i}{N_i} \bigg/ \frac{n_{\max}}{N_{\max}}.$$

2.7 Research Question

Damage as computed by the Cycle Counter or Perseus can be subtracted from the component's Approved Life in order to monitor the remaining life of the component, and remove it from service when this reaches zero. With a view to improving current company processes, this thesis aims to address the following research question:

Using the methods described in this section, can a supervised ML or DL approach be identified that offers a sufficiently robust, verifiable, comprehensive, scalable, fast and accurate means of processing EHM data to determine the extent of damage incurred by surface nodes of a component during real flight missions?

The given input data is EHM data containing a number of real flight missions (Section

3.1), coupled with corresponding Cycle Counter damage values (Section 3.4) for each of the seven features (Section 3.3) as the given output data.

Models based on ML and DL methods will be trained on the complete dataset as well as two subsets (Section 3.6). The aim is to use the trained model to reproduce given output data on corresponding unseen input data with the highest possible accuracy. Further, due to the time and effort involved in gathering the given output data (on which the model is trained), training data.

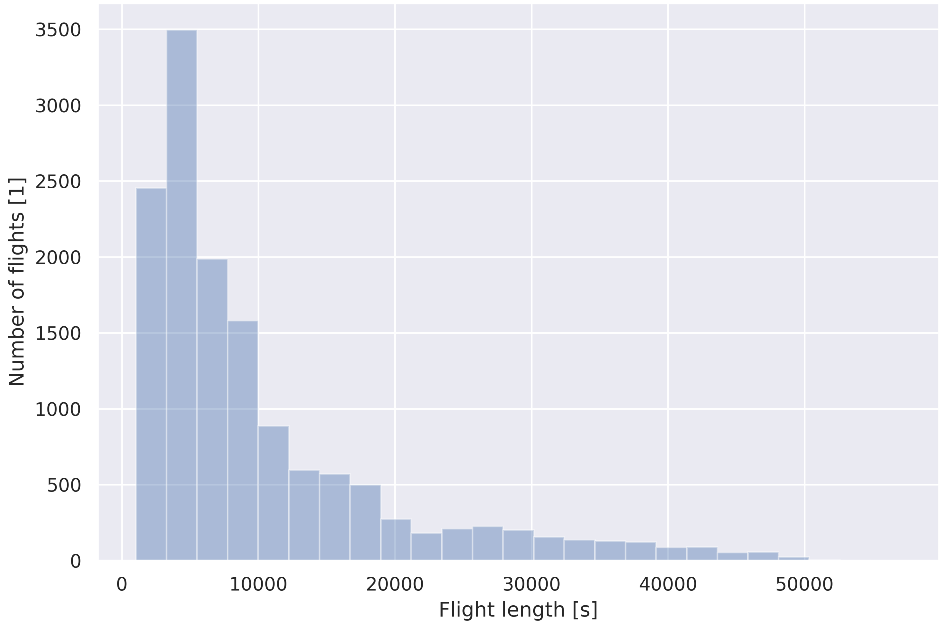

Figure 5: A histogram of the lengths of 14 045 flights

3 Data Analysis

The following section will include an overview and visualisation of the data acquired for the research.

3.1 Engine Health Monitoring Data

The research was carried out on EHM data. This data was recorded by sensors in 231 BR725 engines during a total of 14 045 flights, and returned on a voluntary basis to Rolls-Royce by operators for analysis.

The BR725 is used in the Gulfstream G650, a business jet built for up to 18 passengers. Each G650 has two engines; to minimise the amount of data used, the values used in this thesis were taken from the left engine only.

The flights range in length from 1 013 to 57 062 seconds (approximately 0.28 to 15.85 hours), with a mean length of 10 182.82 seconds and a standard deviation of 9 561.03 seconds (see Figure 5).

Each flight is summarised in a comma-separated values (CSV) file with 216 columns, comprising one timestamp and 215 values per second of recording time.

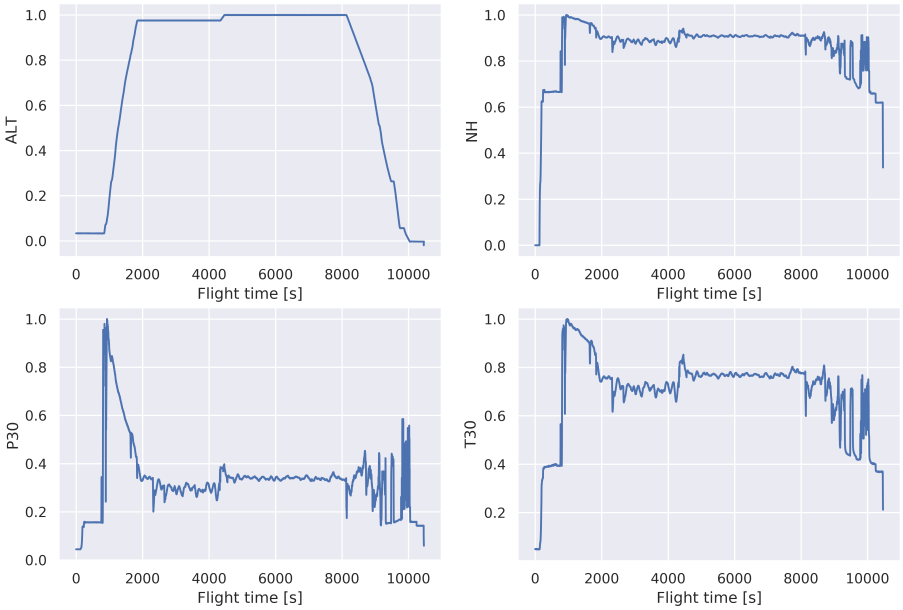

Figure 6: ALT, NH, P30 and T30 of a randomly selected flight. (All parameter values normalised.)

The four flight parameters extracted from these CSV files were altitude (ALT), rotational speed of the high-pressure shaft (NH), and pressure and temperature of air exiting the compressor (P30 and T30, respectively). These values are shown (normalised) for one randomly selected flight in Figure 6.

NH is measured in rotations per minute, and recorded as a percentage of the maximum rotational speed defined for the engine. It therefore takes any value between 0 and 100. ALT is measured in feet; the maximum altitude across all flights was 51 148

Table 1: Summary of flight phases and conditions at which they begin (König, 2018). FM conditions in accordance with Reischl & Müller (2014).

Phase description	Conditions
Preflight	FM = 2
Taxi out	left or right engine is switched on
Take-off	FM = 4
Climb	WOW = 0 intertial vertical speed > 500 ft/min altitude at least 1500 m greater than at take-off
Cruise	FM = 6 altitude greater than 85% of maximum altitude
Descent	FM = 7 or FM = 8 TRA < 20° for both engines Time to destination < 45 min
Reverse thrust	FM = 9
Taxi in	reverse thrust phase ended

feet, or 15 590 metres. P30 is measured in TODO, and ranges between 5.3 and 385.8 across all flights. The units of T30 are degrees Celsius and the values range from -12.0°C to 590.5°C.

3.2 Flight Phases

A flight can be split into several phases: preflight, taxi out, take-off, climb, cruise, descent, reverse thrust and taxi in. These phases were extracted using internal Rolls-Royce software (König, 2018) that combined the flight mode parameter from EHM data (Reischl & Müller, 2014) and custom conditions for optimisation. The conditions are summarised in table 1.

The flight mode (FM) often makes use of the parameter weight on wheels (WOW), a boolean parameter with a value of 1 if the aircraft's weight is supported by its wheels, otherwise 0. Other parameters used for determining FM include ground speed, intertial vertical speed, wing flap angle and thrust lever resolver angle (TRA).

Figure 7 shows the NH plot from the same flight as in Figure 6; dashed vertical lines

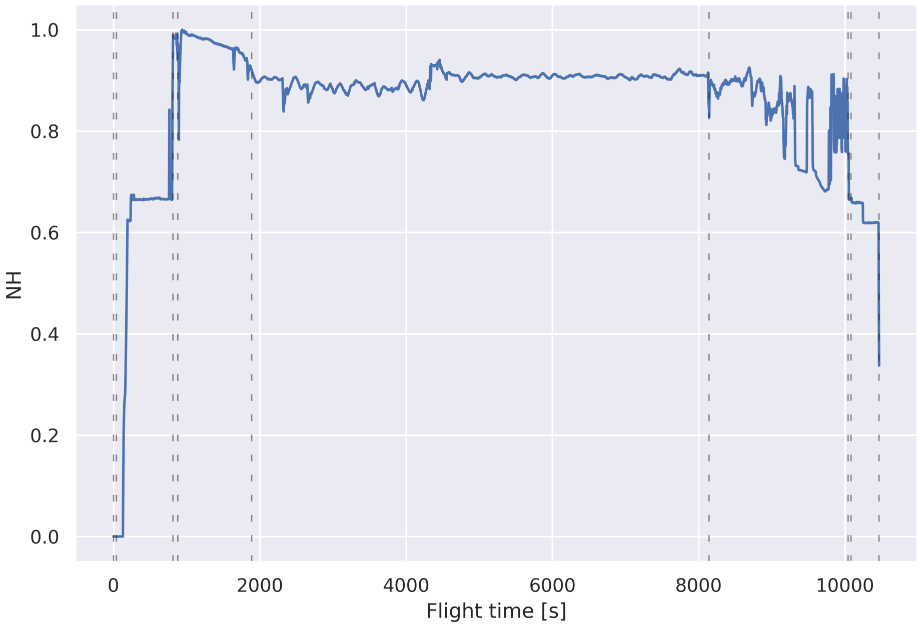

Figure 7: Phase boundaries of NH indicated on the same flight from Figure 6

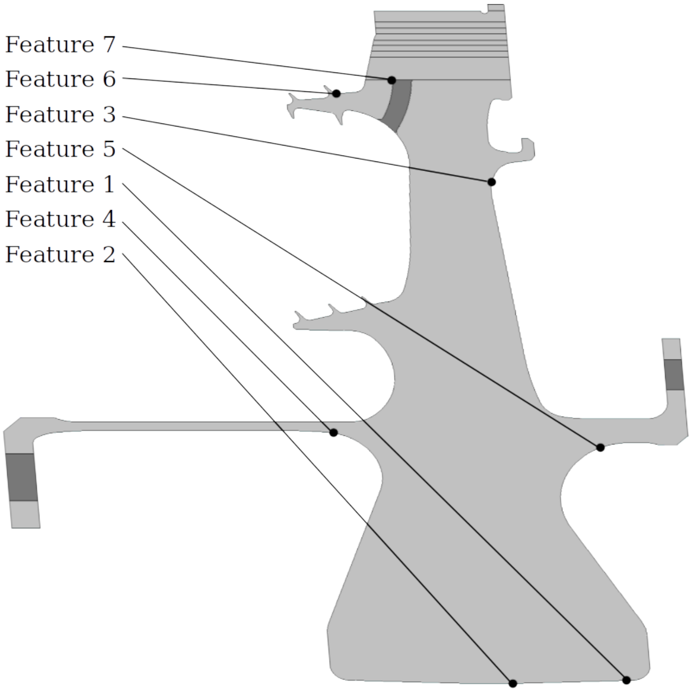

Figure 8: Location of features 1 to 7 on the HPT disk. (The disk was scaled and sheared to protect intellectual property.)

represent phase boundaries.

3.3 Features

Seven features of the HPT disc were selected as points of interest for the research. The features were selected based on TODO.

- 7 features selected based on

- Plot of HPT cross-section and feature locations

- 1/2 NH-driven, 4/5 P30, 6/7 T30 + plots

3.4 Cycle Counter

The EHM data was processed using the company's internal Cycle Counter software to determine the number of cycles consumed by each feature during each flight. Figures 9 and 10 show the distribution of damage data (98 315 values across 7 features from 14 045 flights) computed by the Cycle Counter. For clarity in illustration and comparison, the distributions were split at 2 cycles: Figure 9 contains all damage values between 0 and 2 cycles, with the overwhelming majority of the values (97 868, 99.5%) falling within this range; Figure 10 contains all values above 2 cycles (with a maximum damage of 14.1 cycles), containing only 447 data points, or 0.05%, of the total.

It should be noted that, in Figure 9, there are clear pairwise similarities in the distributions of the pairs mentioned in Section 3.3. This is of no surprise, but could be highly valuable if a supervised machine learning model can be trained to use this (and other) information to extrapolate to further features.

- Non-negligible amount of effort required to obtain this data for supervised learning. Big question: Can model scale to smaller data sets?

3.5 Downsampling

- Required for input layer to MLP

- Also better for scalability of models

3.5.1 (A)PLA

3.5.2 APCA

3.5.3 Data Loss

- Average percentage loss caused by downsampling.

- Will it affect model accuracy?

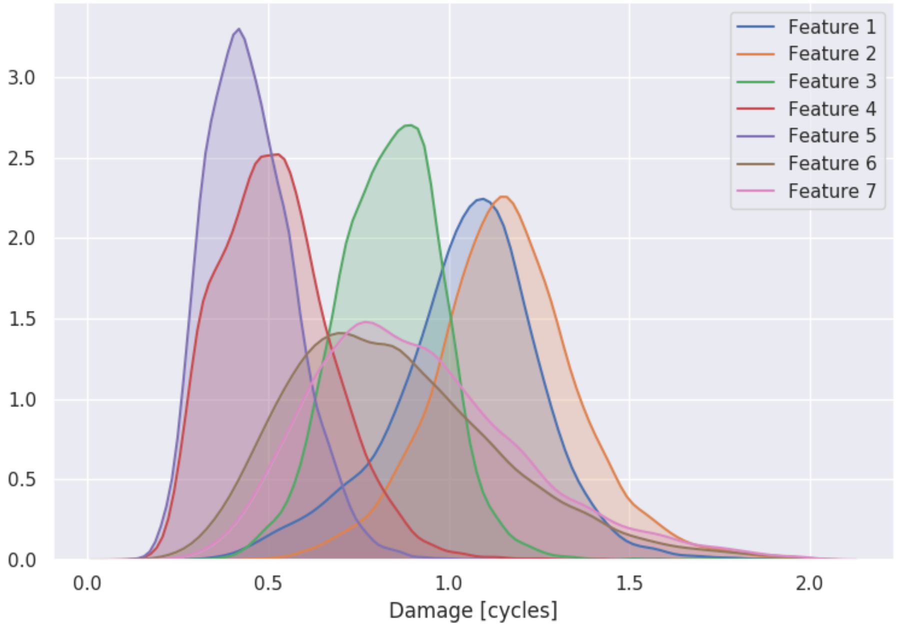

Figure 9: KDE plot displaying the distribution of damage cycles consumed by the seven features over 99.5% of all 14 045 flights (with damage of up to two cycles)

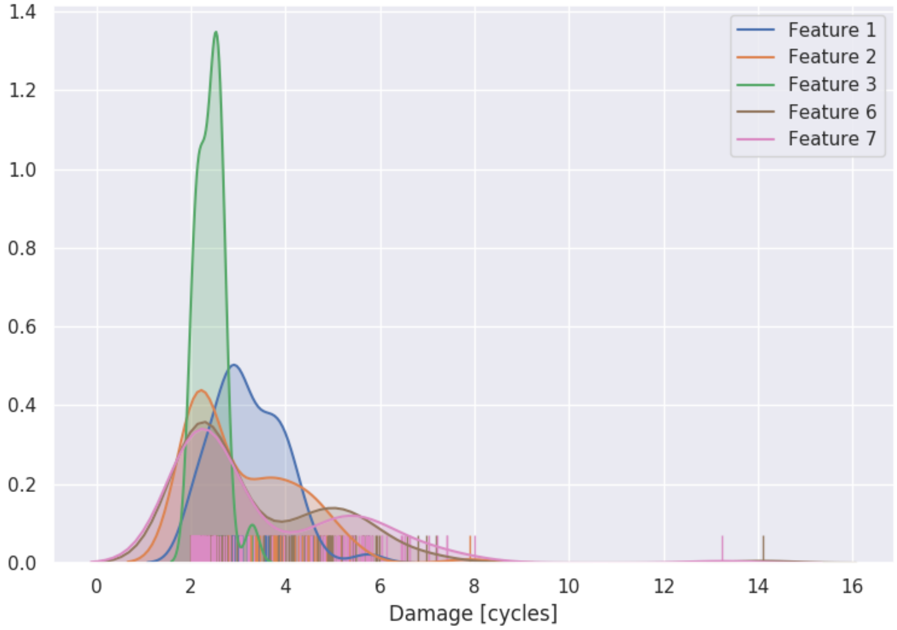

Figure 10: KDE plot displaying the distribution of damage cycles consumed by the seven features over 0.5% of all 14 045 flights (with damage of two cycles and above). The short vertical lines at the base represent individual data points. No flight caused damage of more than 2 cycles in features 4 and 5.

3.6 Dataset Reduction

The full dataset consists of EHM data for 14 045 flights, as well as seven Cycle Counter output values for each flight. Since scalability is an important factor in the research, the dataset was separated into two further datasets of 4 213 and 1 211 flights. (These three datasets will be referred to as ‘complete’, ‘reduced’ and ‘greatly reduced’ datasets, respectively, in the following.)

These datasets were further split into training and validation subsets at a ratio of 3:1. The complete dataset therefore consisted of 10534 training and 3511 validation flights, the reduced dataset 3160 and 1053, and the greatly reduced dataset 909 and 302. The flights numbers in each set and subset were kept constant throughout all models to avoid skewing the data.

4 Model Implementation

To investigate the research question, seven ML and DL models were trained and validated on the three dataset sizes described in Section 3.6.

Two types of supervised problems were considered: regression and classification. Due to the fundamental difference in the output values of models for these two tasks (a decimal number in regression and several binary class predictions for classification), the terms *hit* and *hit rate* were introduced to enable comparison between the models.

For classification models, a hit is simply a correct label prediction. A hit by a regression model was defined as a damage prediction correct within a tolerance of ±0.15 cycles from the Cycle Counter value, often referred to as y_{true} in the following. The hit rate is determined by dividing the number of hits by the total number of predictions. The value of 0.15 is somewhat arbitrary and by no means an acceptable tolerance for a model in production, but in these initial stages is a challenging but achievable goal.

The Python programming language (version 3.7.5) was used for developing the models. For the machine learning methods and MLP models, the `scikit-learn` package (version 0.22.1) was put to use. The CNNs were developed using `Keras` (version 2.3.1) and `tensorflow` (version 1.15.0).

4.1 Polynomial Regression on Key Values

The first model presented is a supervised machine learning method using the `PolynomialFeatures` class from `scikit-learn`, which is based on the polynomial regression method as described in Section 2.3.1.

- supervised ML method

- key value regression: max(), exchange rate (should description of ER go in theory?), taxiout/climb/cruise length

- scikitlearn ‘PolynomialFeatures’ transforms a polynomial regression into linear regression

- usual linear regression performance measure is R^2; for comparability, using MSE and “hits”

4.2 MLP: Key Value Regression

Using the `MLPRegressor` class from `scikit-learn`,

TODO: One model with, one without F1, F4, F6

- Same values as polynomial, plus F1, F3, F7 (3 should have been 4!)

- Some feature vals included as a few are OK to calculate if these improve the results significantly.

4.3 MLP: Time Series Regression

- supervised DL

- time series regression over entire, unfiltered time series

- tentatively expecting good predictions: Due to downsampling, manoeuvres tend to take place at the same points in TS, so shapes can be learned. However, MLP is not time-invariant as each perceptron has its own weights and biases, so if manoeuvres such as in Flight 6014 are to be recognised, MLP not suitable.

4.4 Convolutional Neural Network: Time Series Classification

TODO: How many predictions landed in jeweiliger Klasse??

- Damage for each feature divided into classes of equal size

- Equal size classes necessary to avoid model defaulting to the most common class, to avoid empty classes, and to spread out values close to median to emphasise differences

4.4.1 Four classes

- What bin boundaries?

- Suffers from point in 2.4.2

4.4.2 Four classes with gap

- Essentially binary choice between “good” and “bad” flights.

- High success expected (see visual difference in f3_NH_high_low_dmg_500.png, also violin plot) but with only limited applicability due to training model on only 50% of data.

- Also restricted in scalability by having to separate flights by feature

- More an explorative model due to impracticability

- Point in 2.4.2 not relevant here as binary choice

4.4.3 Ten classes

- Bin boundaries

- Should this be included? No better than TS Regression... Also very similar.

- Less of a classification problem, more a rounded regression problem!

4.5 Convolutional Neural Network: Time Series Regression

- SotA research on UCR archive by hfawaz showed InceptionTime as current frontrunner, marginally in terms of accuracy but significantly in terms of training time.

- Codebase (footnote link) already accepts MTS; minimal changes required to turn this into a regression.

4.5.1 Time Series Regression

By changing the activation function of the output layer from softmax to ReLU, a classifier can be changed into a regressor. Naturally, the input data must follow suit, and the model can only be trained if the given output data is at the very least ordinal.

5 Discussion

6 Conclusion

References

Bagnall, A., Lines, J., Bostrom, A., Large, J. & Keogh, E. (2017, May). The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, *31*(3), 606–660. Retrieved 2020-03-26, from `http://link.springer.com/10.1007/s10618-016-0483-9` doi: 10.1007/s10618-016-0483-9

Bräunling, W. J. (2015). *Flugzeugtriebwerke: Grundlagen, Aero-Thermodynamik, ideale und reale Kreisprozesse, Thermische Turbomaschinen, Komponenten, Emissionen und Systeme.* Springer-Verlag.

Chen, M., Mao, S. & Liu, Y. (2014, April). Big Data: A Survey. *Mobile Networks and Applications*, *19*(2), 171–209. Retrieved 2020-04-16, from `http://link.springer.com/10.1007/s11036-013-0489-0` doi: 10.1007/s11036-013-0489-0

Corran, R. S. J. & Williams, S. J. (2007, April). Lifing methods and safety criteria in aero gas turbines. *Engineering Failure Analysis*, *14*(3), 518–528. Retrieved 2020-03-18, from `http://www.sciencedirect.com/science/article/pii/S1350630706001014` doi: 10.1016/j.engfailanal.2005.08.010

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., … Keogh, E. (2019, November). The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, *6*(6), 1293–1305. (Conference Name: IEEE/CAA Journal of Automatica Sinica) doi: 10.1109/JAS.2019.1911747

Deng, L. & Li, X. (2013, May). Machine Learning Paradigms for Speech Recognition: An Overview. *IEEE Transactions on Audio, Speech, and Language Processing*, *21*(5), 1060–1089. (Conference Name: IEEE Transactions on Audio, Speech, and Language Processing) doi: 10.1109/TASL.2013.2244083

EASA. (2015). *Certification Specifications for Engines (CS-E).* European Union Aviation Safety Agency. Retrieved 2020-03-18, from `https://www.easa.europa.eu/`

`document-library/certification-specifications/cs-e-initial-issue` (Library Catalog: www.easa.europa.eu)

FAA. (2007). *Guidance material for 14 CFR Section 33.75, safety analysis.* Retrieved 2020-03-18, from `http://link.library.in.gov/portal/Guidance-material-for-14-CFR-Section-33.75/ME0hLfglA20/` (Library Catalog: link.library.in.gov)

Fan, W. & Bifet, A. (2013, April). Mining big data: current status, and forecast to the future. *ACM SIGKDD Explorations Newsletter*, *14*(2), 1–5. Retrieved 2020-04-17, from `https://dl.acm.org/doi/10.1145/2481244.2481246` doi: 10.1145/2481244.2481246

Fawaz, H. I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D. F., Weber, J., … Petitjean, F. (2019, September). InceptionTime: Finding AlexNet for Time Series Classification. *arXiv:1909.04939 [cs, stat].* Retrieved 2020-03-18, from `http://arxiv.org/abs/1909.04939` (arXiv: 1909.04939)

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning.* MIT Press. (Google-Books-ID: omivDQAAQBAJ)
\url{http://www.deeplearningbook.org}

Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J. & Seung, H. S. (2000). Digital selection and analogue ampli®cation coexist in a cortex-inspired silicon circuit. , *405*, 5.

Harrington, P. (2012). *Machine learning in action.* Shelter Island, N.Y: Manning Publications Co. (OCLC: ocn746834657)

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L. & Muller, P.-A. (2019, July). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, *33*(4), 917–963. Retrieved 2020-03-18, from `https://doi.org/10.1007/s10618-019-00619-1` doi: 10.1007/s10618-019-00619-1

Jarrett, K., Kavukcuoglu, K., Ranzato, M. A. & LeCun, Y. (2009, September). What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th*

International Conference on Computer Vision (pp. 2146–2153). Kyoto: IEEE. Retrieved 2020-04-26, from `http://ieeexplore.ieee.org/document/5459469/` doi: 10.1109/ICCV.2009.5459469

Kelleher, J. D., Namee, B. M. & D'Arcy, A. (2015). *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. MIT Press. (Google-Books-ID: uZxOCgAAQBAJ)

Kirk, M. (2017). *Thoughtful Machine Learning with Python: A Test-Driven Approach*. "O'Reilly Media, Inc.". (Google-Books-ID: nG3vDQAAQBAJ)

König, J. (2018, December). *BR725Stats*. Rolls-Royce Deutschland Ltd & Co KG.

Krollner, B., Vanstone, B. & Finnie, G. (2010, January). Financial Time Series Forecasting with Machine Learning Techniques: A Survey..

Kyprianidis, K. G. (2011). Future Aero Engine Designs: An Evolving Vision. , 23.

Lee, W., Park, S., Joo, W. & Moon, I.-C. (2018, November). Diagnosis Prediction via Medical Context Attention Networks Using Deep Generative Modeling. In *2018 IEEE International Conference on Data Mining (ICDM)* (pp. 1104–1109). (ISSN: 1550-4786) doi: 10.1109/ICDM.2018.00143

Palmgren, A. (1924). Die Lebensdauer van Kugellagern. *VDI Zeitschrifft*, *68*.

plc, R.-R. (2015). *The Jet Engine* (5th edition ed.). Chichester, West Sussex: Wiley-Blackwell.

Rajaraman, V. (2016, August). Big data analytics. *Resonance*, *21*(8), 695–716. Retrieved 2020-03-18, from `https://doi.org/10.1007/s12045-016-0376-7` doi: 10.1007/s12045-016-0376-7

Reischl, B. & Müller, V.-D. (2014, October). *BR700-725A1-12 G650 EHM SSRD EIS Engine Health Monitoring (EHM) and Engine E-TR0109 09-ISS10.* Rolls-Royce Deutschland Ltd & Co KG.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. Retrieved 2020-04-26, from `http://doi.apa.org/getdoi.cfm?doi=10.1037/h0042519` doi: 10.1037/h0042519

Spittle, P. (2003, November). Gas turbine technology. *Physics Education*, *38*(6), 504–511. Retrieved 2020-03-18, from `http://stacks.iop.org/0031-9120/38/i=6/a=002?key=crossref.5d4ad51e02e416f47faed1970f94d375` doi: 10.1088/0031-9120/38/6/002

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2014, September). Going Deeper with Convolutions. *arXiv:1409.4842 [cs]*. Retrieved 2020-04-17, from `http://arxiv.org/abs/1409.4842` (arXiv: 1409.4842)

VanderPlas, J. (2016). *Python Data Science Handbook.* Retrieved 2020-05-08, from `http://shop.oreilly.com/product/0636920034919.do`

Yang, Q. & Wu, X. (2006, December). 10 CHALLENGING PROBLEMS IN DATA MINING RESEARCH. *International Journal of Information Technology & Decision Making*, *05*(04), 597–604. Retrieved 2020-03-26, from `https://www.worldscientific.com/doi/abs/10.1142/S0219622006002258` doi: 10.1142/S0219622006002258

Zhou & Chellappa. (1988, July). Computation of optical flow using a neural network. In *IEEE 1988 International Conference on Neural Networks* (pp. 71–78 vol.2). doi: 10.1109/ICNN.1988.23914