Forecasting remaining useful life: Interpretable deep learning approach via variational Bayesian inferences

Mathias Kraus[a,*], Stefan Feuerriegel[a]

[a]*ETH Zurich, Weinbergstr. 56/58, 8092 Zurich, Switzerland*

Abstract

Predicting the remaining useful life of machinery, infrastructure, or other equipment can facilitate preemptive maintenance decisions, whereby a failure is prevented through timely repair or replacement. This allows for a better decision support by considering the anticipated time-to-failure and thus promises to reduce costs. Here a common baseline may be derived by fitting a probability density function to past lifetimes and then utilizing the (conditional) expected remaining useful life as a prognostic. This approach finds widespread use in practice because of its high explanatory power. A more accurate alternative is promised by machine learning, where forecasts incorporate deterioration processes and environmental variables through sensor data. However, machine learning largely functions as a black-box method and its forecasts thus forfeit most of the desired interpretability. As our primary contribution, we propose a structured-effect neural network for predicting the remaining useful life which combines the favorable properties of both approaches: its key innovation is that it offers both a high accountability and the flexibility of deep learning. The parameters are estimated via variational Bayesian inferences. The different approaches are compared based on the actual time-to-failure for aircraft engines. This demonstrates the performance and superior interpretability of our method, while we finally discuss implications for decision support.

Keywords: Forecasting, Remaining useful life, Machine learning, Neural networks, Deep learning

Declarations of interest: *none*

[*]Corresponding author.
Email addresses: `mathiaskraus@ethz.ch` (Mathias Kraus), `sfeuerriegel@ethz.ch` (Stefan Feuerriegel)

1. Introduction

Maintenance of physical equipment, machinery, systems and even complete infrastructure represents an essential process for ensuring successful operation. It helps minimizing downtime of technical equipment [1], eliminate the risk thereof [2], or prolong the life of systems [3]. Maintenance is often enforced by external factors, such as regulations or quality management [4]. Yet maintenance burdens individuals, businesses and organizations with immense costs. For instance, the International Air Transport Association (IATA) reported that maintenance costs of 49 major airlines increased by over 3 percent from 2012 to 2016, finally totaling \$15.57 billion annually.[1]

Decision support in maintenance can be loosely categorized according to two different objectives depending on whether they serve a corrective or preemptive purpose.[2] The former takes place after the failure of machinery with the goal of restoring its operations back to normal. Conversely, preemptive maintenance aims at monitoring these operations, so that the time-to-failure can be predicted and acted upon in order to mitigate potential causes and risk factors by, for instance, replacing deteriorated components in advance. Preemptive actions help in reducing downtime and, in practice, promise substantial financial savings, thus constituting the focus of this paper.

Preemptive maintenance is based on estimations of the remaining useful life (RUL) of the machinery. While preventive maintenance makes these forecasts based on human knowledge, predictive maintenance utilizes data-driven models. Different models have been proposed that can be categorized by which input data is utilized (see Section 2 for an overview). In the case of raw event data, the conventional approach involves the estimation of probability density functions. If sensor data is available, the prominent approach draws upon machine learning models [8, 9]. The latter fosters non-linear relationships between sensor observations and RUL estimates, which aid in obtaining more accurate forecasts.

Machine learning models are subject to an inherent drawback: they frequently operate in a black-box fashion [10, 11, 12], which, when providing decision support, directly impedes potential insights into the underlying rules behind their decision-making. However, *interpretability* is demanded for a

[1]International Air Transport Association (IATA). *Airline maintenance cost executive commentary.* URL: `https://www.iata.org/whatwedo/workgroups/Documents/MCTF/MCTF-FY2016-Report-Public.pdf`, accessed April 18, 2019.

[2]Despite the wealth of earlier works on maintenance operations, there is no universal terminology. Instead, the interested reader is referred to Jardine et al. [5], Heng et al. [6], and Si et al. [7] for detailed overviews. We adhere to their terminology.

variety of practical reasons. For instance, practitioners desire to benchmark predictive models with their own expertise, as well as to validate the decision-making rules from machine learning models against common knowledge [13]. Further, managers can identify potential causes of a short machine lifetime and, thus, outline means by which to reduce errors [14]. Moreover, accountability in RUL forecasts is sometimes even required by regulatory agencies, such as, e. g., in aircraft or railroad maintenance [e. g. 15, 16].

Interpretability refers to machine learning models where the decision logic of the model itself is transparent. Notably, the concept of interpretability differs from post-hoc *explainability* that aims for a different objective. Here, a single (or multiple random) forecast is decomposed, thus highlighting potential relationships but without any structural guarantees [17]. That is, explainability takes an arbitrary model as input and, based on it, attempts to unravel the decision logic behind it, but does so only for a local neighborhood of the input rather than deriving its actual structure. Hence, post-hoc explanations are often not reliable, result in misleading outputs and, because of that, the need for interpretable machine learning has been named an important objective for safety-aware applications [18]. By constructing models that are inherently interpretable, practitioners obtain insights into the underlying mechanisms of the model [19]. In keeping with this, we formulate our research objective as follows.

Objective: Forecasting remaining useful life via machine learning with the additional requirement that the model fulfills the definition of "interpretability".

We develop interpretable deep learning models for forecasting RUL as follows: we propose a novel structured-effect neural network that represents a viable trade-off between attaining accurate forecasts and the interpretability from simple distributional estimations. In order to estimate its parameters, we develop an innovative estimation technique based on variational Bayesian inferences that minimize the Kullback-Leibler divergence.[3]

We demonstrate the effectiveness of our approach in terms of interpretability and prediction performance as follows. We utilize the public "Turbofan Engine Degradation Simulation" dataset [20] with sensor measurements from aircraft engines. This dataset is widely referred to as a baseline

[3]Some researchers have raised concerns about the applicability of variational Bayesian inference to neural networks, specifically as alternatives might potentially be more straightforward to optimize. Yet variational Bayesian inferences entail obvious strengths in our setting: in contrast to other approaches, it allows to include prior domain knowledge (as is done in our work when choosing regularization priors).

for comparing predictive models in maintenance and RUL predictions; see e. g., Butcher et al. [21] and Dong et al. [22]. Here the goal is to forecast the remaining useful life until irregular operations, such as breakdowns or failures, take place. The proposed structured-effect neural network outperforms the distribution-based approaches, reducing the forecast error by 51.60 percent. While our approach is surpassed slightly by deep learning, it fulfills the definition of being interpretable, i. e., it maintains the same accountability as the much simpler probabilistic approaches.

The remainder of this paper is structured as follows. Section 2 provides an overview on predicting remaining useful life for preemptive maintenance. Section 3 then introduces our methodological framework consisting of probabilistic approaches, machine learning and the novel structured-effect neural network that combines the desirable properties of both. The resulting performance is reported in Section 4, where we specifically study the interpretability of the different approaches. Finally, Section 5 concludes with a discussion of our findings and implications of our work with respect to decision support.

2. Background

Previous research has developed an extensive range of mathematical approaches in order to improve maintenance and, due to space constraints, we can only summarize core areas related to our work in the following. For detailed overviews, we refer to Heng et al. [6], Liao and Kottig [23], Navarro and Rychlik [24] and Si et al. [7], which provide a schematic categorization of run-to-failure, condition monitoring and predictive methods that estimate the remaining useful life of the machinery. Depending on the underlying approach, the resulting strategy can vary between corrective, responsive, or preemptive maintenance operations. Predictive maintenance, in particular, gives rise to a multitude of variants, e. g., probabilistic approaches and fully data-driven methods that rely upon machine learning together with granular sensor data. The intuition behind inserting sensor measurements into predictive models is that the latter can quantify the environment numerically, the operations and the potential deterioration [25]. The observed quantities can be highly versatile and include vibration, oil analysis, temperature, pressure, moisture, humidity, loading, speed, and environmental effects [7]. As such, sensor measurements are likely to supersede pure condition-based signals in their contribution to overall prognostic capability.

Approach	**Decision variable**	**Input variables**	**Maintenance strategy**	**Operationalization**
Run-to-failure	—	—	Corrective	Service on failure
Condition monitoring	Latent state	Event/sensor data	Responsive	Service when latent state indicates (upcoming) failure
Physics-based RUL models	Physics-based RUL	Simulation models	Preemptive	Service when RUL reaches predefined threshold
Probabilistic RUL models	Population-wide (or conditional) RUL	Event data	Preemptive	Service when RUL reaches predefined threshold
Sensor-based RUL predictions (e. g., proportional hazards model, machine learning)	**System-specific RUL**	**Sensor data**	**Preemptive**	**Service when RUL reaches predefined threshold**

Table 1: Schematic overview of key research streams for using RUL models in maintenance operations. Further hybridizations of approaches exists that are not covered by the categorization.

In order to carry out preemptive measures, one estimates the *remaining useful life* (RUL) and then applies a suitable strategy for scheduling maintenance operations (such as a simple threshold rule that triggers a maintenance once RUL undercuts a safety margin) in a cost-efficient manner [26, 16, 27]. Mathematically, the RUL at time t can be formalized as a random variable Y_t that depends on the operative environment and its past use $X_t, \ldots, X_2, X_1$, i. e.,

$$\mathbb{E}\left[Y_t \mid X_t, \ldots, X_2, X_1\right]. \tag{1}$$

Here the variables $X_1, \ldots, X_t$ can refer to event data tracking past failures [28], numerical quantities tracing the machines condition over time as an early warning of malfunctioning [7], or measurements of its use as a proxy for deterioration [24].

2.1. Probabilistic lifetime models

Probabilistic models utilize knowledge about the population of machinery by learning from the sensor observations of multiple machines. This knowledge is obtained utilizing predefined probability density functions that specify the probability distributions over machinery lifetimes. Mathematically, when X_t is not available, the RUL estimation turns into $\mathbb{E}\left[Y_t \mid X_t, \ldots, X_2, X_1\right] = \mathbb{E}\left[Y_t\right] = \frac{\mathbb{E}\left[t + Y_t\right]}{R(t)}$, where $R(t)$ is the survival function at t. Common choices include exponential, log-logistic, log-normal, gamma, and Weibull distributions [e. g. 6]. We refer to Navarro and Rychlik [24] for a detailed survey. For instance, the Weibull distribution has been found to be effective even given few observations of lifetimes, which facilitates its practical use [29]. Both log-normal and Weibull distributions can be extended by covariates for sensor-data, which we describe below in Section 3.5 but are then constrained to the mathematical structure, rather than flexibility when calibrating a data-driven approach through machine learning.

Probabilistic approaches are common choices as they benefit from straightforward use, direct interpretability and reliable estimates, that are often required in practical applications and especially by the regulatory body. However, a focus is almost exclusively placed on raw event data, thereby ignoring the prognostic capacity of sensor data.

Probabilistic approaches can theoretically be extended to accommodate sensor data, resulting in survival models. Since its initial proposal by Cox [30], the proportional hazards model has been popular for lifetime analysis in general [31] and the estimation of RUL in particular. A key advantage of the proportional hazards model over many other approaches is that the interaction between a number of influencing factors can be easily combined with a baseline function that describes the general lifetime of the machinery. More precisely, the proportional hazards model assumes that the probability estimates consists of two components, namely, a structural effect and random effects described by covariates [7]. As will be discussed later, our structured-effect neural network is built on a similar idea; however, it exploits deep learning to increase the predictive power of the RUL in contrast to the proportional hazards model, which utilizes an exponential model to describe the random effect.

2.2. Machine learning in lifetime predictions

Machine learning has recently received great traction for RUL as the flexibility of these models facilitates a superior prognostic capacity. For instance, linear regression models offer the advantage of high interpretability when predicting RUL. Extensions by regularization yield the lasso and ridge regression, which have been found to be effective for high-dimensional sensor data [32]. To overcome the limitations of linear relationships, a variety of non-linear models have been utilized, including support vector regression [8], random forests [9], and neural networks [33]. We refer to Heng et al. [6] and Si et al. [7] for a detailed overview of the proposed models. However, non-linear models generally fall short in terms of their explanatory power [10].

Even though machine learning demonstrates high predictive power, these models struggle with the nature of sensor data as time series. It is common practice to make RUL estimates based purely on the sensor data at one specific point in time [7]. This simplifies $\mathbb{E}\left[Y_t \mid X_t, \ldots X_1\right]$ to $\mathbb{E}\left[Y_t \mid X_t\right]$, thereby ignoring the past trajectory of sensor measurements. Yet the history of sensor measurements is likely to encode valuable information regarding the past deterioration and usage of machinery. As an intuitive example, a jet engine that experiences considerable vibration might require more frequent check-ups. As a remedy, feature engineering has been proposed in order to aggregate past usage profiles onto feature vectors that are then fed into the machine learning model [34]. Formally, this yields $\mathbb{E}\left[Y_t \mid \phi(X_t, \ldots, X_1)\right]$ or $\mathbb{E}\left[Y_t \mid X_t, \phi(X_{t-1}, \ldots, X_1)\right]$, where the aggregation function ϕ could, for instance, extract the maximum, minimum, or variability from a sensor time series. As a result, the features could theoretically be linked to interpretations but this is largely prohibited by the nature of the machine learning model.

Advances from deep neural networks have only recently been utilized for the prediction of RUL. In Babu et al. [35], the authors apply convolutional neural networks along the temporal dimension in order to incorporate automated feature learning from raw sensor signals and predict RUL. In other works, long short-term memory networks (LSTMs), as a prevalent form of recurrent neural networks, have been shown to perform superior to traditional statistical probability regression methods in predicting RUL [22, 36, 37]. Thereby, the LSTM can make use of the complete sequence of sensor measurements by processing complete sequences with the objective of directly estimating the formula $\mathbb{E}\left[Y_t \mid X_t, \ldots, X_1\right]$ with varying, machine-dependent t. In addition, LSTMs entail a high degree of flexibility, which helps to accurately model highly non-linear relationships. This commonly lowers the forecast error, which further translates into improved maintenance operations.

Deep neural networks are rarely utilized in practical applications for a variety of reasons. Arguably, this is not only because deep neural networks have only recently begun to be used for estimating RUL, but also because they are widely known to be black-box functions with limited to no interpretability. Hence, it is the contribution of this paper to develop a combination of structural predictions and deep learning in order to reach a favorable trade-off between interpretability and prognostic capacity. As points of comparison, we draw upon previous works for RUL predictions, including those concerned with machine learning, feature engineering, and deep learning.

2.3. Verification in machine learning

Interpretability of machine learning is particularly important in mission-critical systems, which requires the development of assessment techniques that reliably identify unlikely types of error [38]. One approach to uncovering cases where the model may be incompatible with the desired behaviour is to systematically search for worst-case results during evaluation [e. g. 39]. Formal verification proves that machine learning models are specification consistent [e. g. 40]. While the field of formal verification has been subject to research, these approaches are impeded by limited scalability, especially in response to modern deep learning systems.

2.4. Explainable vs. interpretable machine learning

Explainable machine learning refers to *post-hoc* explaining predictions without elucidating the mechanisms with which models work. Examples of such post-hoc interpretations are local linear approximation of the model's behavior [e. g. partial dependence plots; see 41] or decompositions of the final prediction into the contribution of each input feature [e. g. SHAP values; see 42]. Another widely applied approach to obtaining explanations is to render visualizations to determine qualitatively what a model has learned [43]. However, explainable machine learning is limited in understanding the underlying process of estimation. Notably, it is also limited to a local neighborhood of the input space or the prediction.

In contrast, interpretable machine learning is to encode an interpretable structure *a priori*, which allows to looking into their mechanisms in order to understand the complete functioning of predictions for all possible input features [44]. Here global relationships are directly encoded in the structure of the model. As such, the relationship in individual features or outcomes for average cases are explicitly modeled. Naturally, linear models have become a prevalent choice for applications in (safety-)critical use cases, where a complete traceability of the model's estimation is inevitable. Hence, the estimation (rather than predictions) can now be compared against prior knowledge or used for obtaining insights.

3. Methods

This research aims at developing forecasting models for the remaining useful life that, on the one hand, obtain a favorable out-of-sample performance while achieving a high degree of interpretability at the same time. Hence, this work contributes to the previous literature by specifically interpreting the relevance of different sensor types and usage profiles in relation to the overall forecast. To date, probabilistic models of failure rates have been widely utilized for predicting the remaining useful life due to their exceptional explanatory power. We thus take the interpretable feature of this approach and develop a method that combines it with the predictive accuracy of deep learning.

We compare the forecasting performance of our structured-effect neural network with the following approaches: (i) naïve empirical estimations, (ii) probabilistic approaches, (iii) traditional machine learning, (iv) traditional machine learning with feature engineering for time series applications, and (v) deep neural networks. All of the aforementioned methods are outlined in the following.

3.1. Naïve empirical estimation of remaining useful life

Naïve empirical estimation of remaining useful life describes the approximation of RUL utilizing past lifetimes of the machinery. Let Z denote the random variable referring to the total lifetime of a machinery, and let $Z_1, \ldots, Z_n$ denote n realizations of this random variable. Then we utilize the mean of these realizations to estimate the total lifetime of a machinery, i. e.,

$$\mathbb{E}[Z] = \frac{1}{n} \sum_{i=1}^{n} Z_i. \tag{2}$$

We can now translate this estimation of the total lifetime into an estimation of the RUL by subtracting the time the machinery has been in use since last being maintained. Let Y_t denote the random variable that describes the RUL of a machinery at time t. Then we estimate Y_t by

$$\mathbb{E}[Y_t] = \mathbb{E}[Z] - t. \tag{3}$$

3.2. Probabilistic lifetime models

In accordance with our literature review, we draw upon two prominent probability density functions P that model the lifetime expectancy of machinery, namely, the Weibull distribution, and the log-normal distribution [e. g. 45]. Let, again, Z denote the random variable referring to the total lifetime of the population. Then the probability density functions of the Weibull distribution and the log-normal distribution at time step $t > 0$ are given by

$$P_{\text{Weibull}}(Z; a, b) = \frac{b}{a}\left(\frac{Z}{a}\right)^{b-1} \mathrm{e}^{-(\frac{Z}{a})^b} \quad \text{and} \tag{4}$$

$$P_{\text{log-normal}}(Z; a, b) = \begin{cases} \frac{1}{\sqrt{2\pi} b Z} \mathrm{e}^{-\frac{(log(Z)-a)^2}{2b^2}}, & Z > 0, \\ 0, & Z \leq 0, \end{cases} \tag{5}$$

respectively, with distribution parameters a and b. All distribution parameters are estimated based on past event data; more precisely, the historical time-spans between failures of the machinery are inserted as the lifetime Z. This allows us to estimate the expected lifetime of the machinery after a maintenance event, as well as the corresponding variance.

The mean value of the different probability density functions *could* provide estimates of the remaining useful life for unseen data observations. However, this would ignore the knowledge that the machine has already functioned over t time steps. Hence, we are interested in the conditional expectation, given that the machine had the last maintenance event t time steps ago. This results in an estimated RUL at time t of

$$\mathbb{E}[Y_t] = \mathbb{E}_{Z \sim P}\left[Z \mid Z > t\right] - t. \tag{6}$$

To compute the previous expression, we draw upon the cumulative distribution function $F(Z; \cdot)$ and the definition of the conditional probability. We then rewrite Equation (6) into

$$\mathbb{E}[Y_t] = \mathbb{E}_{Z \sim P}\left[Z \mid Z > t\right] - t = \mathbb{E}_{Z \sim P}\left[\frac{Z}{1 - F(Z; \cdot)}\right] - t. \tag{7}$$

Unfortunately, there is no (known) closed-form solution to the expected conditional probability of a Weibull distribution. Hence, we utilize Markov chain Monte Carlo to approximate Equation (7) for both, the Weibull distribution and the log-normal distribution, in order to come up with the expected remaining lifespan (conditional on the time of the last maintenance event).

3.3. Traditional machine learning

In the following, let f refer to the different machine learning models with additional parameters w. Then, in each time step t, the machine learning model f is fed with the current sensor data X_t and computes the predicted RUL, given by $\tilde{Y}_t = f(X_t; w)$, such that $Y_t \approx \tilde{Y}_t$. The deviation between the true RUL, Y_t, and the forecast $\tilde{Y}_t$ defines the prediction error that we try to minimize. Hence, the optimal parameters can be determined by an optimization problem

$$w^* = \arg\min_{w} \|Y_t - f(X_t; w)\|. \tag{8}$$

A variety of models f are common in predicting remaining useful life; see the surveys in Heng et al. [6] and Si et al. [7]. We adhere to previous choices and thus incorporate a variety of baseline models that consist of both linear and non-linear models. Linear models include ridge regression, lasso, and elastic net, all of which are easily interpretable and have been shown to perform well on many machine learning tasks with high-dimensional and even collinear features [46]. The set of non-linear baseline models include random forest and support vector regression (SVR).

All models are then fed with two different sets of features: (1) we take the current sensor measurements X_t when predicting the RUL estimate Y_t. However, this approach ignores the trajectory of historic sensor data. (2) As a remedy, we rely upon feature engineering as a means of condensing the past time series into a feature vector, as described in the following.

Feature engineering provides a means by which to encode the past usage of machinery into an input vector for the predictive model. Yet previous research has only little guidance at hand regarding what type of features are most useful. Hence, we adapt the choice of aggregation functions from Mosallam et al. [34], as detailed in Table 2. For instance, vibration is known to accelerate deterioration, but it is unclear whether this is caused by sudden peaks (i. e., minima or maxima), frequent changes (i. e., standard deviation), or a constantly high tremor (i. e., average). Mathematically, each aggregation function ϕ takes a sequence of past sensor measurements $X_t, \ldots, X_1$ as input and then computes a new input feature $\phi(X_t, \ldots, X_1)$. These aggregation functions are necessary to map the complete trajectory onto a fixed, predefined number of features that can be readily processed by the machine learning models.

Aggregation function	**Formula**	**Interpretation**
Max	$\max(X_1, \ldots, X_t)$	Extrema
Min	$\min(X_1, \ldots, X_t)$	Extrema
Mean	$\mu = \frac{1}{t} \sum_{i=1}^{t} X_i$	Average sensor measurement
Range	$\max - \min$	Variability
Sum	$\sum_{i=1}^{t} X_i$	Total signal
Energy	$\sum_{i=1}^{t} X_i^2$	Total signal with focus on peaks
Standard deviation	$\sigma = \sqrt{\frac{1}{t} \sum_{i=1}^{t} (X_i - \mu)^2}$	Variability
Skewness	$\frac{1}{t} \sum_{i=1}^{t} \left(\frac{X_i - \mu}{\sigma} \right)^3$	Symmetry of deviation
Kurtosis	$\frac{1}{t} \sum_{i=1}^{t} \left(\frac{X_i - \mu}{\sigma} \right)^4$	Infrequent extreme deviations
Peak-to-peak	$\frac{1}{n_1} \sum_{i=1}^{n_1} \text{loc max} + \frac{1}{n_2} \sum_{i=1}^{n_2} \text{loc min}$	Bandwith
Root mean square	$\sqrt{\frac{1}{t} \sum_{i=1}^{t} X_i^2}$	Total load focus on peaks
Entropy	$- \sum_{i=1}^{t} P(X_i) \log P(X_i)$	Information signal
Arithmetic mean of power spectral density	$20 \log_{10} \frac{\frac{1}{t} \sum_{i=1}^{t} \lvert \text{fft}(X_i) \rvert}{10^{-5}}$	Frequency of oscillations
Line integral	$\sum_{i=1}^{t-1} \lvert X_{i+1} - X_i \rvert$	Path length
Kalman filter	$Y_t - b - \sum_{i=1}^{p} a_i X_{t-i}$	Unexpected deviation

Table 2: Our feature engineering draws upon the above aggregation functions. This choice is common in predicting remaining useful life [e. g. 34]. Here p refers to an optional parameter specifying the number of lags, which is later set to 50 in accordance with previous research. The expressions loc max and loc min refer to the local maximum and minimum of the inputs.

To determine the best hyperparameter combination in traditional machine learning, we implemented group 10-fold cross-validation in order to minimize the bias associated with random sampling of training and validation data. The group approach also ensures that we do not split maintenance cycles during cross-validation and that the same maintenance cycle is not present in both the training and validation set.

3.4. Recurrent neural network

Recurrent neural networks refer to a special class of deep neural networks that can learn from sequences of varying lengths, rather than a fixed size of feature vector [47]. This is beneficial to our setting, as it allows us to directly inject time series with sensor data into the RNN and predict the remaining useful life from it. The mathematical formalization is as follows: let f_{NN} denote a traditional (or deep) neural network that defines a mapping $[X_t, h_{t-1}] \mapsto h_t$ with hidden states

$h_{t-1}, h_t \in \mathbb{R}^n$ and a suitably chosen dimension n. Then a prediction from a complete sequence can be made via

$$RNN_\Theta = f_{\text{NN}}([X_t, f_{\text{NN}}([X_{t-1}, \ldots f_{\text{NN}}([X_1, \mathbf{0}_n])])]). \tag{9}$$

In other words, the RNN iterates over the sequence, while updating its hidden state h_t, which summarizes the already-seen sequence, similar to an internal state. This recurrent relationship between the states introduces the possibility of passing information onwards from the current state h_t to the next h_{t+1}. Therefore, RNNs can process sequences of arbitrary length, making them capable of utilizing the complete trajectory of sensor data. To illustrate this, Figure 1 presents the processing of sequential data by means of unrolling the recurrent structure.

Different variants of recurrent neural networks have been proposed in earlier research; see Goodfellow et al. [47]. In this work, we choose the long short-term memory from Hochreiter and Schmidhuber [48] because it is capable to keep information over long sequences, and it enjoys widespread use in research and practical applications [49, 50, 51]. For deep neural networks, we reduce the computational runtime for hyperparameter tuning and instead follow conventional guidelines, whereby a random sample of the training data (10 %) serves for validation.

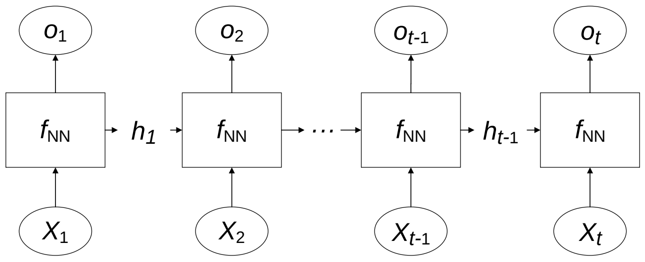

Figure 1: Recurrent neural network that recursively applies the same simple neural network f_{NN} to the input sequence $X_1, \ldots, X_t$ with outputs $o_1, \ldots, o_t$. The states $h_1, \ldots, h_{t-1}$ encode the previous sequence into a fixed-size feature vector.

3.5. Proposed structured-effect neural network

3.5.1. Model specification

We now propose our structured-effect model. This approach enforces a specific structure that lends to intuitive interpretation. More precisely, it combines non-parametric approaches for modeling the expected RUL through a probabilistic density function with the flexibility of machine learning in order to incorporate sensor measurements and thus capture the heterogeneity from machine-specific deterioration processes.

The idea of building structured models loosely resembles earlier research efforts related to proportional hazards models [30] that present a popular choice in lifetime analysis. This class of survival models decomposes its estimations into components referring to the baseline function and a function of covariates: the baseline specifies the general lifetime across all machines via the same function $\lambda(t)$. The latter further assumes machine-specific effects through additional covariates that describe the random effects. Our structured-effect neural network follows a similar intuition, as it assumes a population-wide general lifetime common across all machines and further sensor-based deviations in order to model the within-machine heterogeneity due to the different usage profiles.

Our structured-effect model splits the estimated remaining useful life into three components, namely, a non-parametric baseline, a covariate-based prediction, and a recurrent component which specifically incorporates the historic trajectory of sensor measurements. These components help in explaining the variance among the different machine lifetimes and, for this purpose, we again draw upon the history of sensor measurements $X_t, \ldots, X_1$. Let $\lambda(t)$ denote the non-parametric part with the explicit probabilistic lifetime model and let further RNN_Θ refer to a recurrent neural network (such as a long short-term memory) with weights Θ. Then the prediction of the structured-effect neural network $SENN_\Theta(t; X_t, \ldots, X_1)$ follows the form

$$SENN_\Theta(t; X_t, \ldots, X_1) = \underbrace{\lambda(t)}_{\substack{\text{Non-parametric component}\\ \text{with explicit lifetime model}}} + \underbrace{\beta^T X_t}_{\substack{\text{Linear component}\\ \text{with current condition}}} + \underbrace{RNN_\Theta(X_t, \ldots, X_1, t)}_{\substack{\text{Recurrent component}\\ \text{with deep neural network}}} \tag{10}$$

with coefficients β. Model variations are discussed later in Section 4.5.

While our model follows a similar intuition behind the proportional hazards model in decomposing the prediction, it also reveals clear differences, as it introduces a recurrent neural network that allows for considerably higher flexibility in modeling the variance and even incorporates the *complete* sequence of sensor measurements and not just a simple vector of covariates. Moreover, the specific way of our model formulation entails a set of further advantages. On the one hand, it circumvents again the explicit need for feature engineering. On the other hand, it achieves a beneficial trade-off between interpretability of non-parametric approaches and the flexibility of non-linear predictions from sensor data. Here the deep neural network needs to explain a considerably smaller variance compared to an approach based solely on a neural network, thereby facilitating the estimation of the network weights. To this end, practitioners can decompose the prediction into a population-wide baseline and machine-specific heterogeneity, based on which they can explicitly quantify the relative contribution of each components through the corresponding coefficients. As such, one can identify reasons why the remaining useful life attains a certain value (e. g., a negative value from the recurrent component indicates a strong deterioration over time) or one can attribute deterioration to unexpected behavior. Moreover, the proposed approach is highly extensible and can easily be generalized to other parameterizations or domains.

We later experiment with different variations of the structured-effect model. These differ in the choices with which we specify the different components. First, we adhere to conventional approaches in predictive maintenance [24] by assuming that the lifetimes follow either a conditional Weibull or a conditional log-normal distribution. That is, we obtain

$$\lambda(t) = \mathbb{E}_{Z \sim \text{Weibull}(a,b)}[Z \mid Z > t] - t \quad \text{and} \quad \lambda(t) = \mathbb{E}_{Z \sim \text{log-normal}(a,b)}[Z \mid Z > t] - t \tag{11}$$

with distribution parameters a and b. Thus, the first component is identical to the probabilistic lifetime models that we utilize as part of our benchmarks. Second, the linear component can either be fed directly with X_t or, alternatively, one could also apply feature engineering to it, i. e., giving $\phi(X_t, \ldots, X_1)$. The benefit of the latter is that we again obtain a linear structure where one can assess the relevance of individual predictors by looking at the coefficients. Here we further assume a linear combination as used in ordinary least squares and, as an extension, introduce priors, so that we yield a regularization, where the coefficients in the linear component are estimated via the least absolute shrinkage operator (lasso). This performs implicitly variable selection in the linear component as some coefficients are directly set to zero [52]. Third, the recurrent neural network is implemented via a long short-term memory as this represents the state-of-the-art in sequence learning [47].

3.5.2. Model estimation through variational Bayesian inferences

We now detail how we estimate the parameters inside the structured-effect neural network. We refer to θ as the combined set of unknown parameters and X as the overall dataset including all sensor measurements. Then the objective is to determine the optimal parameters

$$\theta^* = \underset{\theta}{\arg\max}\; P(\theta \,|\, X). \tag{12}$$

We solve the previous optimization problem through a variational Bayesian method. The predominant reason for this choice over traditional optimization is that the latter would merely give point estimates of the different parameters, whereas variational Bayesian inferences yield quantifications of uncertainty. For instance, this allows us to obtain confidence assessments concerning the relative importance of the different components and thus facilitates the interpretability of our approach.

In our model estimation, we treat all parameters as latent variables with a pre-defined prior distribution and, subsequently, maximize the overall likelihood of the parameters according to the following procedure. That is, utilizing Bayes' theorem, Equation (12) is rewritten to

$$P(\theta \mid X) = \frac{P(X \mid \theta)\, P(\theta)}{P(X)} = \frac{P(X \mid \theta)\, P(\theta)}{\int P(X \mid \theta)\, P(\theta)\, \mathrm{d}\theta}. \tag{13}$$

As a result, the denominator can be computed through sampling methods, with the most prominent being Markov chain Monte Carlo (MCMC). However, MCMC methods are computationally expensive as the runtime scales exponentially with the dimensions of θ. Thus, this algorithm becomes intractable for large-scale or high-dimensional datasets. As a remedy, we propose the use of variational Bayes for approximating the posterior distributions. We derive a variational lower bound, called ELBO, for our structured-effect neural network in Appendix A.

3.5.3. Estimation parameters

In our experiments, we optimize the *SENN*-model by utilizing the Adam optimizer with learning rate 0.005 and all other parameters set to the default values. All implementations are performed in Python utilizing the probabilistic programming library "pyro" (`http://pyro.ai/`). Code for reproducibility is available online.[4].

As part of our computational experiments, we later draw upon the following architectures of the structured-effect neural network: (1) we assume the non-parametric component to follow a Weibull or log-normal prior distribution, where the underlying distribution parameters are modeled as informative normal prior distributions. Mathematically, this is given by $a \sim \mathcal{N}(a_{\text{empirical}}, 1)$ and $b \sim \mathcal{N}(b_{\text{empirical}}, 1)$. (2) The linear component is modeled such that the coefficients stem from normal prior distributions (i. e., as used in ordinary least squares). This is formalized by $\beta_i \sim \mathcal{N}(0, 10)$, where we allow for a wider standard deviation to better handle variations in the relative influence of the predictors. As an alternative, we also implement weakly informative prior distributions (i. e., Laplace priors). The latter enforce a regularization similar to the least absolute shrinkage operator in

[4]See `https://github.com/MathiasKraus/PredictiveMaintenance`

the sense that certain coefficients are set exactly to zero in order to perform implicit variable selection and come up with a parsimonious model structure. (3) The recurrent component is implemented as a long short-term memory network with two layers containing 100 and 50 neurons, respectively. To reduce computational costs, we follow common approaches and utilize the trajectory of the previous 50 sensor values at all time steps. All weights in the network are implemented as variational parameters that follow a Gaussian prior with standard deviation of 1. Utilizing Equation (A.10), we optimize the three components simultaneously.

4. Computational experiments

4.1. Dataset

For reasons of comparability, all computational experiments are based on the "Turbofan Engine Degradation Simulation" dataset, which is widely utilized as a baseline for comparing predictive models in maintenance and RUL predictions; see e. g., Butcher et al. [21] and Dong et al. [22]. The objective is to predict the RUL (measured in cycles) based on sensor data from 200 aircraft engines.[5] More specifically, it includes measurements from 21 sensors. Unfortunately, however, the exact name of each sensor is sanitized. In addition, the dataset comes with a pre-determined split into a training set (100 engines) and a test set (also 100 engines). The average RUL spans 82.30 cycles with a standard deviation of 54.59. Moreover, half of the engines experience a failure within 77 cycles, while only 25 % exceed 118 cycles.

4.2. Prediction performance for remaining useful life

The prediction results for all models are listed in Table 3. Here we report the mean absolute error, as it represents a widely utilized metric for this dataset [53]. The benefit of this metric is that practitioners can easily translate the forecast error into a number of cycles that would serve as a security margin. The table also compares two different feature sets for traditional machine learning, i. e., on which we use only the sensor measurements from the current time step or on which we additionally apply aggregation functions to the sensors as part of feature engineering.

The empirical RUL in the first row reflects the performance of our naïve benchmark when using no predictor (i. e., predicting the average RUL of the machines). The following conditional expectations

[5]The specific dataset of this study can be downloaded from `https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/`, accessed April 18, 2019.

are based on the Weibull and log-normal distribution, that result in improvements of 39.17 % and 38.32 %, respectively. Among the traditional machine learning models, we find the lowest mean absolute error when using the random forest, which yields an improvement of 35.08 % compared to the log-normal-based conditional expectation. Thereby, our results identify a superior performance through the use of feature engineering for the majority of traditional machine learning models. Recurrent neural networks outperform traditional machine learning. In particular, the LSTM yields the overall lowest mean absolute error, outperforming the random forest with feature engineering by 37.12 % and the empirical RUL by 60.51 %.

The structured-effect neural networks outperform traditional machine learning. Utilizing a log-normal distribution along with feature engineering yields an improvement of 25.44 % compared to the best traditional machine learning model. Thereby, feature engineering accounts for 11.91 % of the improvement, strengthening the assumption that feature engineering of sensor data facilitates the prediction of RUL.

Method		MAE	Forecast comparison (t-statistic)				Forecast comparison (P-value)			
			Best baseline	Best machine learning	Best LSTM	Best structured-effect LSTM	Best baseline	Best machine learning	Best LSTM	Best structured-effect LSTM
Baselines without sensor data										
Empirical RUL		45.060	8.286	24.933	16.974	18.865	0.468	0.486	0.778	0.642
Conditional expectation (Weibull)		27.794	0.288	8.309	9.615	8.018	0.293	0.462	0.666	0.457
Conditional expectation (log-normal)		**27.409**	—	4.420	15.269	10.285	—	**0.464**	**0.669**	**0.451**
Traditional machine learning										
Ridge regression		19.193	−6.789	1.438	5.270	2.768	0.012*	0.139	0.450	0.322
Ridge regression (with feature engineering)		18.382	−8.029	0.427	3.297	2.778	0.010*	0.132	0.343	0.399
Lasso		19.229	−7.015	0.766	6.577	4.145	0.015*	0.222	0.500	0.401
Lasso (with feature engineering)		18.853	−7.842	0.550	5.949	2.324	0.014*	0.293	0.432	0.390
Elastic net		19.229	−7.276	0.990	5.190	2.829	0.015*	0.222	0.500	0.401
Elastic net (with feature engineering)		18.245	−9.055	0.458	4.244	2.572	0.009**	0.132	0.297	0.245
Random forest		17.884	−4.927	0.058	5.909	4.487	0.006**	0.102	0.240	0.198
Random forest (with feature engineering)		**17.793**	−9.495	—	2.924	3.263	**0.006****	—	**0.236**	**0.180**
SVR		18.109	−7.321	0.240	5.756	3.976	0.011*	0.129	0.288	0.230
SVR (with feature engineering)		21.932	−4.706	3.081	9.440	3.740	0.092*	0.310	0.583	0.531
Recurrent neural networks										
LSTM		**11.188**	−11.596	−4.981	—	−1.441	**0.000*****	**0.000*****	—	**0.003****
Structured-effect neural networks										
Distribution	*Linear component*									
Weibull	None	15.862	−11.266	−1.015	4.424	1.825	0.000***	0.066*	0.255	0.200
Weibull	Regularized	17.433	−8.617	−0.213	3.536	2.746	0.000***	0.094*	0.261	0.220
Weibull	Feature engineering	13.392	−8.526	−2.595	1.352	0.068	0.000***	0.004**	0.144	0.134
Weibull	Regularized feature engineering	14.989	−10.918	−1.579	1.710	1.381	0.000***	0.049*	0.284	0.183
log-normal	None	15.061	−6.294	−1.420	1.627	1.702	0.000***	0.057*	0.261	0.198
log-normal	Regularized	16.319	−8.200	−0.779	4.582	1.334	0.000***	0.094*	0.310	0.211
log-normal	**Feature engineering**	**13.267**	−13.620	−2.912	1.764	—	**0.000*****	**0.000*****	**0.142**	—
log-normal	Regularized feature engineering	14.545	−11.331	−1.736	3.293	0.651	0.000***	0.038*	0.252	0.162

Significance level: * 0.1, ** 0.01, * 0.001**

Table 3: Comparison of prediction performance over remaining useful life across different model specifications. Here we specifically report whether the models only utilize sensor measurements from the current time step or whether aggregation functions have been applied to it as part of feature engineering. Consistent with earlier works [53], the mean absolute error (MAE) is given. The best-performing model in each panel is highlighted in bold. Additionally, we perform t-tests between each model and the best performing model from each of the four categories. The t-tests are based on the MAE of the forecasted RUL to show that improvements are at a statistically significant level.

4.3. Forecast decomposition for RUL predictions

We now demonstrate how the proposed structured-effect model achieves accountability over its RUL forecasts. That is, we leverage the linear model specification and compute the estimated values for each summand in Equation (10) when making a RUL prediction. Yet the model can still adapt to non-linearity since the neural network can absorb the variance that cannot be explained by the other components.

Figure 2 illustrates the interpretability of the RUL forecasts for an example engine. More specifically, we can understand how predictions are formed by decomposing the forecasts from the structured-effect model into three components – namely, the probabilistic RUL model, the linear combination of sensor measurements, and an additional neural network – as follows:

1. As we can see, the distribution-based lifetime component accounts for a considerable portion of the forecast. The maximum values in the example exceed 100, which is considerably higher than the maximum value computed by the recurrent component. The component reaches this value when making a prediction after around 50 cycles and, with each subsequent usage cycle, lowers the estimated remaining useful life. Notably, it is identical across all engines as it encodes the prior knowledge before considering the engine-specific deterioration process.

2. The second component specifies a linear combination of sensor measurements, which allows the predictions to adapt to the specific usage profiles of individual engines and explains the within-engine and within-time variability. It thus no longer yields a smooth curve but rather an engine-specific pattern. Formally, this component refers to $\beta^T X_t$ and, in order to determine the relevance of sensor i, we simply interpret the coefficients in the vector β.

3. While the previous linear component still achieves full accountability over its forecasts, we now introduce the final component for modeling the remaining noise. Here we draw upon (deep) neural networks, as they are known to effectively model non-linearities. However, we thus lose the explanatory power for this component, as neural networks largely operate in a black-box fashion. In our example, we see that the recurrent part entails a non-linear curve but takes higher values in later cycles. This indicates that a linear combination is not always sufficient for making predictions and, as a remedy, the structured-effect model can benefit from additional non-linear relationships and from accumulating the usage profile over time.

   Notably, the magnitude of the recurrent component is much smaller than the magnitude of the other components. This is beneficial, as the SENN attributes most of the explained variance to other, interpretable model components. Methodologically, it is likely to be based on the following: at timestep t, the SENN makes prediction of the RUL from the current sensor data X_t, and the trajectory of sensor data $X_t, X_{t-1}, X_{t-2}, \ldots, X_1$. As shown in Table 3, X_t is highly informative for estimating RUL and, by following stochastic gradient descent, it optimizes in the direction where the loss function decreases the most (i. e. in the direction of both the distribution-based and the linear component). Only after optimizing the the interpretable components, the model updates the recurrent component to further push predictive performance via non-linear mappings.

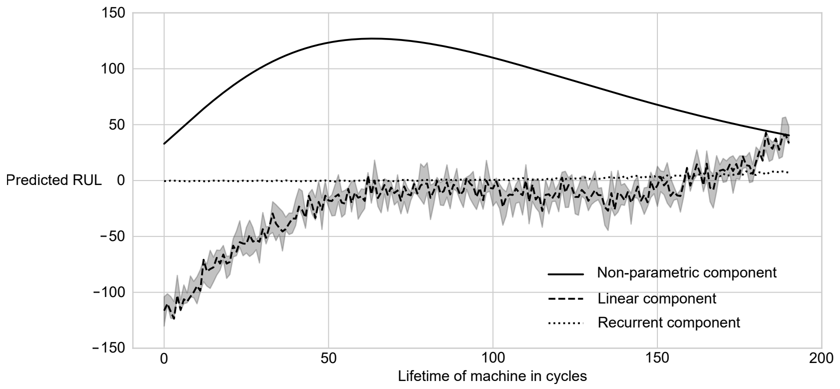

Figure 2: This plot visualizes the RUL predictions made by the structured-effect LSTMs based on a log-normal and normal priors in the linear component for an example engine. It decomposes the forecasts using the structured-effect model into three components that facilitate interpretations of how predictions are formed. (1) The distribution-based lifetime component contributes a considerable portion of the overall forecasts, as it is well suited to model the overall nature of the remaining useful life. This part is identical across all engines. (2) The sensor measurements introduce a variability that adapts to the specific usage profile of an engine. This component originates from a Bayesian linear model and we can trace the forecast back to individual sensors. (3) The recurrent neural network introduces a non-linear component that operates in black-box fashion.

We can further compute the fraction of variance explained by the different components relative to the overall variance of the actual RUL values. Thereby, we quantify the contribution of each component to the overall forecast. Accordingly, this is defined by

$$1 - \frac{\sum_t \left(\tilde{Y}_t - \tilde{\Psi}_t\right)^2}{\sum_t \left(\tilde{Y}_t - \frac{1}{T}\sum_t \tilde{Y}_t\right)^2}, \tag{14}$$

where T denotes the total number of observations and $\tilde{\Psi}_t$ is the prediction of the component under study. Accordingly, we obtain a score of 0.175 for the non-parametric part, 0.408 for the linear component, and 0.064 for the recurrent component. This matches our expectations and, once more, highlights the overall importance of the distribution-based part of the overall forecast, as well as the role of the neural network in modeling secondary variations.

4.4. Estimated parameters

It is common practice to compute parameters in predictive models as point estimates [46], while our optimization technique based on variational Bayesian inferences allows for uncertainty estimates.

This yields a key benefit, since we can validate our confidence in the structural components of the model by studying the posterior distribution of the parameter estimates. Figure 3 depicts these parameters specifying the Weibull and log-normal distribution inside the structured-effect models.

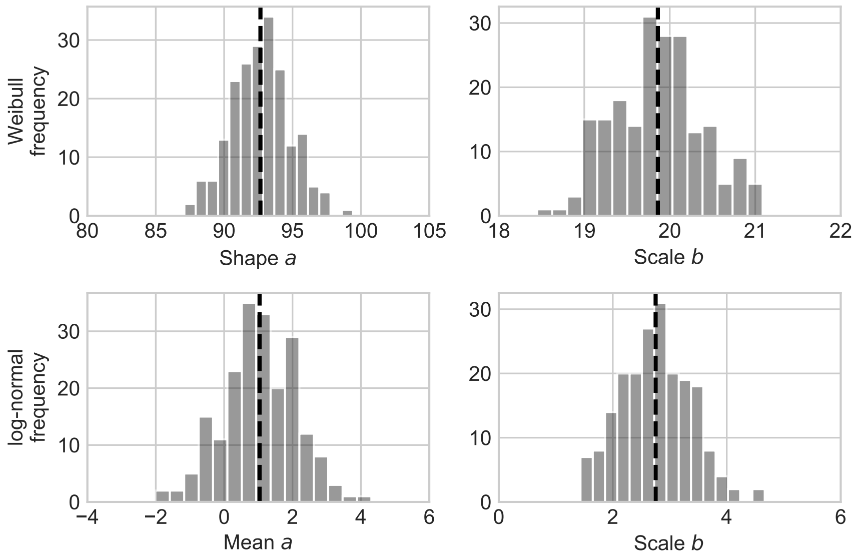

Figure 3: These histograms illustrate the posterior distribution of the estimated parameters (i. e., shape, mean and scale) inside the structured-effect neural network (with log-normal structure and normal priors in the linear component). Here the estimations are compared for both distributions, namely, the Weibull and log-normal distributions. Altogether, the posteriors quantify the uncertainty of the estimated parameters.

Table 4 further reports the posterior distribution of the coefficients β_i from the linear component of the structured-effect neural network (shown is the SENN model based on a log-normal and normal priors for β_i). These measure the effect size, i. e., how a change in a sensor measurement affects the forecast. We see that the confidence regions for the different coefficients vary considerably. For reasons of comparability, we further report the standardized coefficients $\beta_i \operatorname{var}(\beta_i)/\operatorname{var}(Y_1, \ldots, Y_t)$, which correct for the variance of the predictor, as well as the outcome [54]. As a result, this value allows us to rank variables by their importance.

Sensor	**Mean estimate**	**Standard deviation**	**Standardized coefficient**
X_9	-33.169	0.498	-16.506
X_{12}	49.721	0.250	12.440
X_{21}	44.932	0.258	11.601
X_7	48.154	0.230	11.073
X_{11}	-24.622	0.357	-8.796
X_{20}	42.850	0.184	7.880
X_{14}	-22.540	0.317	-7.155
X_4	-16.183	0.275	-4.447
X_{15}	-13.934	0.297	-4.145
X_2	-12.446	0.316	-3.931
X_6	19.678	0.159	3.126
X_3	-7.851	0.318	-2.494
X_{17}	-9.951	0.208	-2.066
X_8	-4.121	0.367	-1.511
X_{16}	-0.273	2.105	-0.574
X_{13}	-1.424	0.348	-0.495
X_{19}	0.234	2.028	0.474
X_{10}	0.126	1.900	0.239
X_{18}	-0.095	1.936	-0.184
X_1	-0.070	1.937	-0.135
X_5	0.001	1.993	0.002

Table 4: Reported here are the posterior estimates of the effect size as measured by the coefficients β_i inside the linear component of the structured-effect neural network. The coefficients entail direct interpretations (similar to ordinary least squares) as to how a certain percentage of change in a sensor measurement affects the RUL prediction. In addition, standardized coefficients are reported, as they allow for the ranking of variables by importance.

4.5. Model variations

We experimented with alternative specifications of our structured-effect neural network as follows.

First, we extended the neural network by an additional weighting factor γ. This yields a component $\gamma\, RNN_\Theta$. However, it resulted in an inferior performance in all of our experiments due to severe overfitting.

Second, we experimented with a two-stage estimation approach. Here we first optimized the non-parametric component and the linear component by traditional gradient descent. Afterwards, we optimized the recurrent component against the residuals from the first stage. This approach is generally easier to train as there are fewer parameters in each stage. Yet we found an inferior performance as compared to the proposed *SENN*: the mean absolute error increased to 16.209 This is possibly owed to the fact that it prevents information sharing between the different components. Details are reported in Appendix B.

5. Discussion

5.1. Implication for decision support

Estimates of the remaining useful life can facilitate decision support with the objective of replacing deteriorated components and thus mitigating potential risks and failures that generally result in increased costs. Our approach to predicting remaining useful life is thus of direct relevance to practitioners. According to a McKinsey report, the use of accurate prediction models for RUL as a cornerstone for predictive maintenance can typically reduce the downtime of machinery by 30 % to 50 % and, at the same time, increase the overall life of machines by 20 % to 40 %.[6] By knowing the exact time-to-failure, companies can plan maintenance ahead of time and, therefore, make preparations for efficient decision support. As a result, even small improvements in predictive power translate into substantial operational cost savings.

As a direct implication for management, this research shows that forward-looking predictive analytics is capable of heavily influencing the way decision support in maintenance operations is conducted. However, predictive models are most powerful when fed by a large number of predictors (i. e. sensors) that describe the condition of the machinery and the environmental effects that influence the system. Therefore, managers should encourage the implementation of additional sensors to further improve accuracy when forecasting the time-to-failure. Moreover, investments in artificial intelligence are oftentimes necessary for the majority of firms who have not yet taken their first step into the age of deep learning.

5.2. Implications for the use of analytics

Trajectories of sensor data accumulate relevant information regarding the past usage profile of the machinery and, thereby, facilitate a prognosis regarding the risk of failure. Mathematically, this results in the objective of finding a mapping $f : [X_1, \ldots, X_t] \mapsto Y_t$ that is not dependent on the current time step t and thus utilizes a time series with historic measurements of arbitrary length in order to infer a prediction from it. The task can be accomplished by a special type of deep learning – namely, recurrent neural networks – as these networks can sequentially process past measurements and store the processed knowledge in their hidden layers. Even though the benefits are obvious, the

[6]McKinsey (2017). Manufacturing: Analytics unleashes productivity and profitability. URL: `https://www.mckinsey.com/business-functions/operations/our-insights/manufacturing-analytics-unleashes-productivity-and-profitability`, last accessed on April 18, 2019.

use of such networks in decision support systems research remains scarce with few exceptions [e. g. 55, 56, 57].

Deep learning is often believed to require extensive amounts of data in order to be successful. However, our approach, based on variational Bayesian estimation, represents a viable alternative that can overcome this limitation. Instead of vast quantities of input data, it advocates domain knowledge that is explicitly encoded in a structural model. In our case, we already know the approximate shape of the predicted variable and can incorporate this via a probability density function into our structural part of the model. As a result, the predetermined structure can be fitted fairly easily with variational inference and thus presents a path towards encoding domain knowledge into deep neural networks. The structured effect reduces the variance and thus makes it easier to describe the remaining variance with a neural network.

5.3. Implications from interpretable forecasts

Our approach contributes to interpretability of deep learning. Here we remind the reader of the difference between explainability and interpretability in machine learning [17, 19, 18]. Explainability merely allows a post-hoc analysis of how predictions were computed in a local neighborhood. In contrast, interpretability presents a stronger notion: it requires machine learning models to attain complete transparency of their decision logic. Thus, we contribute to a novel approach for interpretable machine learning to decision support, that can eventually benefit (safety-)critical application fields where accountable models are required.

The high degree of interpretability of our approach reveals further implications. In practice, gaining insights into the estimated RUL aids engineers in identifying potential risks and weak spots when designing machinery. For instance, a high coefficient for a sensor measuring moisture could encourage designers to improve the sealing of a given piece of machinery. By shedding light on the prediction process, structured-effect neural networks enable novel conclusions regarding the relevance of each sensor.

Decision support as a discipline takes the demands of all stakeholders into account. With regard to the latter, managers, for instance, need to understand the decision-making of automated systems. However, this requirement is not fulfilled by recent trends in advanced analytics and especially deep learning, as these mostly operate in a black-box fashion [e. g. 47]. As a remedy, our structured-effect neural network shows improvements in predictive performance as compared to traditional machine learning, while also allowing for a high degree of interpretability. Table 5 compares the stylized characteristics of our structured-effect neural network to other approaches.

Method	**Predictive performance**	**Interpretability**	**Non-linearities**	**Estimation**
Probabilistic models	Poor (but reliable as no variance is associated with it)	Good (often used along a simple threshold)	Poor (or rather constrained by how well outcomes follow a distribution)	Good (when using sampling over analytic forms)
Linear machine learning	Fair (regularization, especially, can yield parsimonious models and reduces the risk of overfitting)	Good (as coefficients directly quantify the effect size)	Poor (often not regarded or only interaction terms or predefined transformations such as logit)	Good (closed-form solution for ordinary least squares; optimization problems for regularization)
Non-linear machine learning	Good (still regarded as the benchmark against which other models have to compete)	Poor (with the exception of certain approaches, e. g., random forests, that rank variable importance but still don't yield accountability of forecasts)	Good (can adapt well to subgroups, non-linear response curves and interactions)	Fair (often efficient estimations, but without uncertainty quantification)
Deep learning	Good (given sufficient training data)	Poor	Good (even when taking sequences as input)	Poor (challenging hyperparameter tuning)
Structured-effect neural network	Good (theoretically identical to deep learning)	Good (full accountability of the structured effect)	Good (included but only confined to the variance that cannot be explained by the structured effect)	Fair (time-consuming sampling but less prone to unfavorable hyperparameters)

Table 5: Stylized characteristics of different models in machine learning. Here we extend the categorization from Hastie et al. [46, p. 351] to include deep learning and our structured-effect neural network.

Sensors have always been an important part of predictive maintenance, as they allow to monitor and to adjust small changes so that small problems do not turn into big problems [e. g. 58]. Many different sensors monitoring different measurements can be the key to better understanding processes and preventing early failures and consequent downtime. However, complex relationships between potentially large number of sensors and the effect on machinery demands for advanced, non-linear modeling of the remaining useful life. Thus, to fully exploit the information obtained from sensors, interpretability is of great use. Our structured-effect neural network bridges the gap between these key specifications.

5.4. Limitations and potential for future research

Recently, dropout as a Bayesian approximation has been proposed as a simple, yet efficient means to obtain uncertainty estimates for neural networks [59]. This approach leads to models with fewer parameters, which generally facilitates optimization. Further, computationally costs for optimization are lower, compared to the costs when utilizing variational Bayesian inference. However, Bayesian approximation via dropout does not provide uncertainty estimates for coefficients in our model. Additionally, Bayesian approximation via dropout comes at the cost of not being capable of including prior information about the coefficients into the model. As the latter is particularly important for predictive maintenance where expert knowledge is inevitable, we decided to utilize variational Bayesian inference.

5.5. Concluding remarks

Decision support as a field has developed a variety of approaches to improve the cost efficiency of maintenance, especially by predicting the remaining useful life of machinery and linking operational decision-making to it. Common approaches for predictive maintenance include statistical models based on probability density functions or machine learning, which further incorporates sensor data. While the former still serves as widespread common practice due to its reliability and interpretability, the latter has shown considerable improvements in prediction accuracy.

This research develops a new model that combines both advantages. Our suggested structured-effect neural network achieves accountability similar to simple distribution-based RUL models as its primary component, as well as a linear combination of sensor measurements. The remaining variance is then described by a recurrent neural network from the field of deep learning, which is known for its flexibility in adapting to non-linear relationships. For this purpose, all parameters are modeled as latent variables and we propose variational Bayesian inferences for their estimation in order to optimize the Kullback-Leibler divergence. Our findings reveal that our structured-effect neural network outperforms traditional machine learning models and still allows one to draw interpretable conclusions about the sources of the deterioration process.

A. Derivation of ELBO for structured-effect neural network

Our suggested approach draws upon variational Bayesian methods and approximates the true posterior via a variational distribution $Q_\lambda(\theta) \approx P(\theta \,|\, X)$. Here $Q_\lambda(\theta)$ refers to a family of distributions that is indexed by λ and, hence, our optimization problem translates into finding the optimal λ^* along with the corresponding distribution Q_{λ^*}. The following theorems state the mathematical definition of λ^* and introduce a tractable approximation.

Theorem A.1 *The optimal λ^* is given by*

$$\lambda^* = \arg\min_{\lambda} \ \mathbb{E}_{Q_\lambda}[\log Q_\lambda(\theta)] - \mathbb{E}_{Q_\lambda}[\log P(X, \theta)] + \log P(X). \tag{A.1}$$

Proof. The fit between the variational distribution $Q_\lambda(\theta)$ and the posterior distribution $P(\theta \,|\, X)$ can be measured by the Kullback-Leibler divergence. Hence, we yield

$$\lambda^* = \arg\min_{\lambda} \ KL(Q_\lambda(\theta) \,||\, P(\theta \,|\, X)). \tag{A.2}$$

Inserting the definition of the Kullback-Leibler divergence results into

$$\begin{aligned} \lambda^* &= \arg\min_{\lambda} \mathbb{E}_{Q_\lambda} \, [\log Q_\lambda(\theta)] - \mathbb{E}_{Q_\lambda}[\log P(\theta \,|\, X)] && \text{(A.3)} \\ &= \arg\min_{\lambda} \mathbb{E}_{Q_\lambda} \, [\log Q_\lambda(\theta)] - \mathbb{E}_{Q_\lambda}[\log P(X, \theta)] + \log P(X). && \text{(A.4)} \end{aligned}$$

□

Unfortunately, Theorem A.1 is intractable, as it depends on the marginal likelihood of the model, $\log P(X)$. Therefore, the following theorem derives an approximation for the marginal likelihood of the model.

Theorem A.2 *The marginal likelihood of the model $\log P(X)$ can be approximated by the evidence lower bound, $ELBO(\lambda)$, i. e.,*

$$\log P(X) \geq \mathbb{E}_{Q_\lambda}[\log P(X, \theta)] - \mathbb{E}_{Q_\lambda}[\log Q_\lambda(\theta)] = ELBO(\lambda). \tag{A.5}$$

Proof. Utilizing Jensen's inequality, it holds that

$$\begin{aligned} \log P(X) &= \log \int P(X, \theta) \, \mathrm{d}\theta = \log \int P(X, \theta) \frac{Q_\lambda(\theta)}{Q_\lambda(\theta)} \, \mathrm{d}\theta = \log \mathbb{E}_{Q_\lambda} \left[\frac{P(X, \theta)}{Q_\lambda(\theta)} \right] && \text{(A.6)} \\ &\geq \mathbb{E}_{Q_\lambda} \left[\log \frac{P(X, \theta)}{Q_\lambda(\theta)} \right] = \mathbb{E}_{Q_\lambda}[\log P(X, \theta)] - \mathbb{E}_{Q_\lambda}[\log q(\theta)]. && \text{(A.7)} \end{aligned}$$

□

Theorem A.3 *The optimal λ^* can be approximated by*

$$\lambda^* = \arg\max_{\lambda} ELBO(\lambda). \tag{A.8}$$

Proof. From Theorem A.1 and Theorem A.2, it immediately follows that

$$\lambda^* = \arg\min_{\lambda} \; \log P(X) - ELBO(\lambda). \tag{A.9}$$

As $\log P(X)$ is constant with respect to λ, the value λ^* can be approximated by maximizing $ELBO(\lambda)$.

□

In order to optimize $ELBO(\lambda)$, we utilize gradient descent with the gradients defined by

$$\nabla_\lambda ELBO(\lambda) = \nabla_\lambda \mathbb{E}_{Q_\lambda}[\log P(X,\theta)] - \mathbb{E}_{Q_\lambda}[\log q(\theta)] \tag{A.10}$$

$$= \mathbb{E}_{Q_\lambda}[\nabla_\lambda \log q(\theta)\,(\log P(X,\theta) - \log q(\theta))]. \tag{A.11}$$

We further utilize Monte Carlo integration to obtain the estimates of the $ELBO(\lambda)$ and the gradient.

B. Two-stage estimation

Analogous to our *SENN*, we chose the non-parametric component λ to follow a log-normal prior. The underlying distribution parameters were modeled as normal prior distributions, i. e., $a \sim \mathcal{N}(a_{\text{empirical}}, 1)$ and $b \sim \mathcal{N}(b_{\text{empirical}}, 1)$. The linear component β was modeled such that the coefficients stem from normal prior distributions,i. e., $\beta_i \sim \mathcal{N}(0, 10)$. The recurrent component RNN_Θ was implemented as a long short-term memory network with two layers containing 100 and 50 neurons, respectively. Formally, the estimation is specified by as follows:

- Stage 1: $\lambda^*, \beta^* = \arg\max_{\lambda,\beta} \; P(\lambda, \beta \,|\, X)$,

- Stage 2: $RNN^*_\Theta = \arg\max_{RNN_\Theta} \; P(RNN_\Theta \,|\, X, \lambda^*, \beta^*)$.

References

[1] B. Heidergott, T. Farenhorst-Yuan, Gradient estimation for multicomponent maintenance systems with age-replacement policy, Operations Research 58 (2010) 706–718.

[2] H. Groenevelt, L. Pintelon, A. Seidmann, Production lot sizing with machine breakdowns, Management Science 38 (1992) 104–123.

[3] A. Dogramaci, N. M. Fraiman, Replacement decisions with maintenance under uncertainty: An imbedded optimal control model, Operations Research 52 (2004) 785–794.

[4] H. L. Lee, M. J. Rosenblatt, Simultaneous determination of production cycle and inspection schedules in a production system, Management Science 33 (1987) 1125–1136.

[5] A. K. Jardine, D. Lin, D. Banjevic, A review on machinery diagnostics and prognostics implementing condition-based maintenance, Mechanical Systems and Signal Processing 20 (2006) 1483–1510.

[6] A. Heng, S. Zhang, A. C. Tan, J. Mathew, Rotating machinery prognostics: State of the art, challenges and opportunities, Mechanical Systems and Signal Processing 23 (2009) 724–739.

[7] X.-S. Si, W. Wang, C.-H. Hu, D.-H. Zhou, Remaining useful life estimation: A review on the statistical data driven approaches, European Journal of Operational Research 213 (2011) 1–14.

[8] M. Baptista, S. Sankararaman, I. P. de Medeiros, C. Nascimento, H. Prendinger, E. M. Henriques, Forecasting fault events for predictive maintenance using data-driven techniques and ARMA modeling, Computers & Industrial Engineering 115 (2018) 41–53.

[9] M. Seera, C. P. Lim, S. Nahavandi, C. K. Loo, Condition monitoring of induction motors: A review and an application of an ensemble of hybrid intelligent models, Expert Systems with Applications 41 (2014) 4891–4903.

[10] L. Breiman, Statistical modeling: The two cultures, Statistical Science 16 (2001) 199–231.

[11] H. Jang, A decision support framework for robust R & D budget allocation using machine learning and optimization, Decision Support Systems 121 (2019) 1–12.

[12] H. S. Subramania, V. R. Khare, Pattern classification driven enhancements for human-in-the-loop decision support systems, Decision Support Systems 50 (2011) 460–468.

[13] D. Delen, H. Zaim, C. Kuzey, S. Zaim, A comparative analysis of machine learning systems for measuring the impact of knowledge management practices, Decision Support Systems 54 (2013) 1150–1160.

[14] K. L. Reifsnider, S. W. Case, Damage tolerance and durability of material systems, Wiley Interscience, New York, NY, 2002.

[15] Y. Liu, B. Stratman, S. Mahadevan, Fatigue crack initiation life prediction of railroad wheels, International Journal of Fatigue 28 (2006) 747–756.

[16] N. Papakostas, P. Papachatzakis, V. Xanthakis, D. Mourtzis, G. Chryssolouris, An approach to operational aircraft maintenance planning, Decision Support Systems 48 (2010) 604–612.

[17] Z. C. Lipton, The mythos of model interpretability, Communications of the ACM 61 (2018) 36–43.

[18] C. Rudin, Please stop explaining black box models for high stakes decisions, in: Conference on Neural Information Processing Systems, 2018.

[19] Y. Lou, R. Caruana, J. Gehrke, G. Hooker, Accurate intelligible models with pairwise interactions, in: SIGKDD International Conference on Knowledge Discovery and Data Mining, 2013, pp. 623–631.

[20] A. Saxena, K. Goebel, C-mapss data set, NASA Ames Prognostics Data Repository (2008).

[21] J. B. Butcher, D. Verstraeten, B. Schrauwen, C. R. Day, P. W. Haycock, Reservoir computing and extreme learning machines for non-linear time-series data analysis, Neural networks 38 (2013) 76–89.

[22] D. Dong, X.-Y. Li, F.-Q. Sun, Life prediction of jet engines based on LSTM-recurrent neural networks, in: Prognostics and System Health Management Conference, IEEE, 2017.

[23] L. Liao, F. Kottig, Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction, IEEE Transactions on Reliability 63 (2014) 191–207.

[24] J. Navarro, T. Rychlik, Comparisons and bounds for expected lifetimes of reliability systems, European Journal of Operational Research 207 (2010) 309–317.

[25] M. Dong, D. He, Hidden semi-Markov model-based methodology for multi-sensor equipment health diagnosis and prognosis, European Journal of Operational Research 178 (2007) 858–878.

[26] M. J. Kim, V. Makis, Joint optimization of sampling and control of partially observable failing systems, Operations Research 61 (2013) 777–790.

[27] S. Plitsos, P. P. Repoussis, I. Mourtos, C. D. Tarantilis, Energy-aware decision support for production scheduling, Decision Support Systems 93 (2017) 88–97.

[28] D. Ghosh, R. Sharman, H. Raghav Rao, S. Upadhyaya, Self-healing systems: survey and synthesis, Decision Support Systems 42 (2007) 2164–2185.

[29] M. Mazhar, S. Kara, H. Kabernick, Remaining life estimation of used components in consumer products: Life cycle data analysis by Weibull and artificial neural networks, Journal of Operations Management 25 (2007) 1184–1193.

[30] Cox, Regression models and life-tables, Journal of the Royal Statistical Society 34 (1972) 187–220.

[31] Y. Wang, S. Wang, Y. Fang, P. Y. Chau, Store survival in online marketplace: An empirical investigation, Decision Support Systems 56 (2013) 482–493.

[32] S. Zihajehzadeh, E. J. Park, Regression model-based walking speed estimation using wrist-worn inertial

sensor, PLOS ONE 11 (2016) e0165211.

[33] A. Riad, H. Elminir, H. Elattar, Evaluation of neural networks in the subject of prognostics as compared to linear regression model, International Journal of Engineering & Technology 10 (2010) 52–58.

[34] A. Mosallam, K. Medjaher, N. Zerhouni, Nonparametric time series modelling for industrial prognostics and health management, The International Journal of Advanced Manufacturing Technology 69 (2013) 1685–1699.

[35] G. S. Babu, P. Zhao, X.-L. Li, Deep convolutional neural network based regression approach for estimation of remaining useful life, in: Database Systems for Advanced Applications, volume 9642, Springer, 2016, pp. 214–228.

[36] Y. Wu, M. Yuan, S. Dong, L. Lin, Y. Liu, Remaining useful life estimation of engineered systems using vanilla LSTM neural networks, Neurocomputing 275 (2018) 167–179.

[37] S. Zheng, K. Ristovski, A. Farahat, C. Gupta, Long short-term memory network for remaining useful life estimation, in: IEEE International Conference on Prognostics and Health Management, IEEE, 2017, pp. 88–95.

[38] W. Xiang, P. Musau, A. A. Wild, D. M. Lopez, N. Hamilton, X. Yang, J. Rosenfeld, T. T. Johnson, Verification for machine learning, autonomy, and neural networks survey, arXiv preprint arXiv:1810.01989 (2018).

[39] A. Athalye, N. Carlini, D. Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, in: International Conference on Machine Learning, 2018.

[40] G. Singh, T. Gehr, M. Mirman, M. Püschel, M. Vechev, Fast and effective robustness certification, in: Conference on Neural Information Processing Systems, 2018, pp. 10802–10813.

[41] M. T. Ribeiro, S. Singh, C. Guestrin, Why should i trust you? explaining the predictions of any classifier, in: SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1135–1144.

[42] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Advances in Neural Information Processing Systems, 2017, pp. 4765–4774.

[43] L. van der Maaten, G. Hinton, Visualizing data using t-SNE, Journal of Machine Learning Research 9 (2008) 2579–2605.

[44] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, B. Yu, Interpretable machine learning: Definitions, methods, and applications, arXiv preprint (2019).

[45] P. J. Vlok, J. L. Coetzee, D. Banjevic, A. K. S. Jardine, V. Makis, Optimal component replacement decisions using vibration monitoring and the proportional-hazards model, Journal of the Operational Research Society 53 (2002) 193–202.

[46] T. Hastie, R. Tibshirani, J. H. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed., Springer, New York, NY, 2009.

[47] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, Cambridge, MA, 2017.

[48] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (1997) 1735–1780.

[49] T. Fischer, C. Krauss, Deep learning with long short-term memory networks for financial market predictions, European Journal of Operational Research 270 (2018) 654–669.

[50] M. Kraus, S. Feuerriegel, A. Oztekin, Deep learning in business analytics and operations research: Models, applications and managerial implications, arXiv preprint (2018).

[51] S. Srivastava, S. Lessmann, A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data, Solar Energy 162 (2018) 232–247.

[52] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society. Series B (Methodological) 58 (1996) 267–288.

[53] E. Ramasso, Investigating computational geometry for failure prognostics, International Journal of Prognostics and Health Management 5 (2014) 1–18.

[54] J. Bring, How to standardize regression coefficients, The American Statistician 48 (1994) 209–213.

[55] J. Evermann, J.-R. Rehse, P. Fettke, Predicting process behaviour using deep learning, Decision Support Systems 100 (2017) 129–140.

[56] M. Kraus, S. Feuerriegel, Decision support from financial disclosures with deep neural networks and transfer learning, Decision Support Systems 104 (2017) 38–48.

[57] N. Mahmoudi, P. Docherty, P. Moscato, Deep neural networks understand investors better, Decision Support Systems 112 (2018) 23–34.

[58] J. Rabatel, S. Bringay, P. Poncelet, Anomaly detection in monitoring sensor data for preventive maintenance, Expert Systems with Applications 38 (2011) 7003–7015.

[59] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: International Conference on International Conference on Machine Learning, 2016, pp. 1050–1059.