Predicting Remaining Useful Life with Sparse Measurement Data

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 59-10 (2025) 2736–2741

Predicting Remaining Useful Life with Sparse Measurement Data ⋆

Jonas Karlsson * **Alexander Karlsson** **
Ebru Turanoglu Bekar *** **Sunith Bandaru** ****

* *School of Informatics, University of Skövde, Skövde, Sweden (e-mail: jonas.karlsson@his.se)*
** *School of Informatics, University of Skövde, Skövde, Sweden (e-mail: alexander.karlsson@his.se)*
*** *Department of Industrial and Materials Science, Chalmers University of Technology, Gothenburg, Sweden (e-mail: ebrut@chalmers.se)*
**** *School of Engineering Science, University of Skövde, Skövde, Sweden (e-mail: sunith.bandaru@his.se)*

Abstract: Predictive maintenance is a central concept in the shift towards Industry 4.0. Accurately estimating the remaining useful life of a machine, or a machine component, is an important aspect of predictive maintenance. Deep learning models have previously been applied to this task with success. However, these models may not perform well for cases where training data is sparse. In these situations, the model should also provide some degree of uncertainty about its prediction to instill trust in the user. Hence, predictive models should accurately estimate their own uncertainty, in addition to providing correct predictions. In this paper, we propose up-sampling of sparse ballbar measurement data in order to generate adequate samples to train and evaluate deep neural networks. The inference is conducted with three different types of models, Monte Carlo Dropout, variational inference, and deep ensemble. The approaches are compared based on point prediction accuracy, and uncertainty quantification quality. It is found that both Monte Carlo Dropout and deep ensemble perform well in regards to predictive accuracy, with the deep ensemble consistently resulting in the best calibrated uncertainty estimation.

Keywords: Remaining useful life, ball-bar system, deep neural network, Bayesian deep learning, predictive maintenance.

1. INTRODUCTION

The manufacturing industry is currently in a shift towards a more connected environment known as *Industry 4.0*. In this new stage, Predictive Maintenance (PdM) (Lee et al., 2006) has been identified as filling a key role (Zonta et al., 2020) by allowing for both more sustainable manufacturing, reducing costs, and increasing productivity (Achouch et al., 2022; Zonta et al., 2020). The increased connectivity and data gathering in factories has been a critical factor in enabling PdM. With this data on hand, smart systems can learn about and anticipate when machine faults are likely to happen. This allows for maintenance to be carried out before a critical link in the production line suffers a breakdown.

A common way to predict how long a given machine can run before suffering a fault is by estimating its Remaining Useful Life (RUL) (Si et al., 2011). Being able to predict a fault in advance reduces reactive maintenance, enabling service personnel to schedule repairs during planned downtimes. PdM also minimizes downtimes, as the machine will only have to be taken off-line during the repair, with less time spent on waiting on the arrival of personnel or spare parts. Indeed, having a machine run to failure can be costly, as the stop of a critical machine may cause issues for the entire production line.

Compared to the last couple of decades, the amount of sensor data that is available to the manufacturers has increased drastically. This enables smart systems to learn about, and predict, when faults are going to occur in industrial equipment. However, there are still challenges with training Deep Learning (DL) models in cases where historical data is sparse. Additionally, condition monitoring tests may cause interrupts in production. This can be labor intensive, especially when manually conducted. This creates a problem where diagnostic data from machines

⋆ The authors would like to thank the Advanced and Innovative Digitalization Program funded by VINNOVA for their funding of the research project TPdM-Trustworthy Predictive Maintenance (Grant No. 2022-01710), within which this study has been conducted. We specifically would like to thank Volvo GTO Skövde for their guidance and support. We are particularly grateful to the domain experts who generously contributed their time and expertise to this study.

10.1016/j.ifacol.2025.09.460

can be both sparse, and not uniformly distributed in time. For example, a machine may be subject to more tests right before or after a breakdown, while machines that work within expected parameters may go multiple months between measurements. This creates a challenge for DL models, as they commonly require large amounts of data in order to be performant (Sarker, 2021).

Providing an estimation of the model's uncertainty has been identified as an avenue for increasing the trust of a DL model (Kompa et al., 2021). This has also been applied in the case of PdM (Chen et al., 2022). For PdM, accurate uncertainty estimation is important as human operators will act on the predictions given by the model. A model that is unaware of its uncertainty may elicit distrust, and can slow adoption of the technology. This is particularly pertinent when the quantity of training data is low. As we cannot count on the Neural Network (NN) being able to accurately and exhaustively interpret all ranges of data that it will see in a real-life setting.

Thus, the aim of this paper is twofold. First, we want to evaluate the feasibility of interpolating sparse time-series data to train DL models. Secondly, we want to compare the performance of three commonly used methods for obtaining probabilistic predictions from DL models on real-world data.

In this paper, we provide two novel contributions:

- A generalizable approach for processing sparse and sporadic Renishaw QC20 ballbar (Čep et al., 2018) measurement data in a way that makes it possible to use for training DL models. This is done by interpolating between extracted features using the `tsfresh` (Christ et al., 2018) library. To the best of our knowledge, time-series data have not been extracted from ballbar data in this manner before. We also show how manifold learning-based visualization of such kind of data can guide the user in designing and applying DL models. We then validate the utility of the generated data by using it to successfully train DL models to predict RUL.
- We use the dataset to compare the predictive performance and uncertainty estimation quality between three different types DL methods, namely, Monte Carlo Dropout (MC Dropout) (Gal and Ghahramani, 2016), Variational Inference (VI) (Blei et al., 2017), and a deep ensemble (Hansen and Salamon, 1990). We provide empirical evidence that DL can perform accurate inference for real-world use cases within PdM.

It is important to note that the primary aim of this study is not to benchmark against other models, but to explore uncertainty quantification techniques in DL. While DL models have been successfully applied to PdM, their performance can suffer when training data is sparse, which is a common challenge in real-world cases. To address this, we use the interpolated dataset to assess the predictive performance and uncertainty estimation quality of various techniques within a fully real-world PdM scenario. This ensures that our analysis remains centered on evaluating the effectiveness of uncertainty estimation methods in handling sparse data.

2. RELATED WORK

Multiple DL architectures have been applied on PdM data. For example the Long Short-Term Memory (LSTM) network (Nguyen et al., 2022; Youness and Aalah, 2023), or the Convolutional Neural Network (CNN) (Benker et al., 2021; de Pater and Mitici, 2022) architecture. Likewise, these models have been trained with a wide array of probabilistic methods; like deep ensemble (Basora et al., 2025), MC Dropout (de Pater and Mitici, 2022), and VI (Benker et al., 2021).

MC Dropout was proposed by Gal and Ghahramani (2016) as a method for obtaining probabilistic predictions from a NN model. This can almost be seen as a *free lunch* method of obtaining an uncertainty estimation for an otherwise deterministic model as it incurs no extra training complexity. Dropout has previously been used as a regularization method to prevent overfitting (Hinton et al., 2012) by randomly disabling, or dropping, neurons in the NN during training. MC Dropout extends this notion by keeping the dropout active during inference, resulting in slight variations of the predictions depending on what neurons are dropped. A prediction distribution can then be set up by running inference multiple times. Since inference is usually much quicker than training a NN, this is an attractive and scalable option for uncertainty estimation.

VI (Blei et al., 2017) represents the weights of a NN as being drawn from a distribution. Instead of sampling model weights from the posterior distribution like Hamiltonian Monte Carlo (HMC) or No U-Turn Sampler (NUTS) (Hoffman et al., 2014), it re-frames the sampling problem into an optimization problem. This is done by substituting the intractable posterior distribution for another, simpler, distribution. The optimization step is then to minimize the Kullback-Leibler divergence in order to make the substituted distribution as close as possible to the true posterior distribution. This allows for faster training of NNs compared to sampling with HMC, although a limitation of the method is that we will never know the distance to the true posterior.

An ensemble model (Hansen and Salamon, 1990) aggregates multiple individual models in order to increase the total performance and generalizability. This has been successfully implemented with NN (Ganaie et al., 2022), and can act as a third point of comparison to the methods described above. Like MC Dropout and VI, an ensemble model will yield a distribution of predictions, one for each model in the ensemble.

In related literature for methods applied to the C-MAPSS dataset (Saxena et al., 2008), a common processing step is to include historical data when predicting the current RUL (de Pater and Mitici, 2022; Benker et al., 2021; Youness and Aalah, 2023). Indeed, including previous feature inputs may expose a trend in the data, leading to increased model performance. This is done by applying a time window over the data, moving once per new measurement. A drawback of this method is that, for a time window of size N, the first $N-1$ data points in the dataset will effectively be lost, as they are used to fill out the first window.

To evaluate the point prediction accuracy of RUL values, metrics like Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are commonly used (de Pater and Mitici, 2022; Benker et al., 2021; Nguyen et al., 2022). However, as expressed by de Pater and Mitici (2022), only comparing point scores disregards the variance and sharpness of the predictions. To alleviate this problem, the tightness of the distribution can be measured with the Continuous Ranked Probability Score (CRPS) metric. In addition to sharpness, a desired feature of the uncertainty is that it should be calibrated. In the PdM context, this has previously been measured by using the Reliability Score (RS) (de Pater and Mitici, 2022).

3. DATA

The data used in this paper includes measurements from Renishaw ballbar tests (Čep et al., 2018). These tests are performed by fastening the ballbar, a telescopic linear sensor, between a fixed and a moving point on the machine. The machine then conducts a circular test by moving around the fixed point in a circle, clockwise (CW) and counter clockwise (CCW). The deviations from the perfect circle are recorded by the ballbar. The data provided by the Renishaw software contains a time series of measurement values (circularity deviations), and some metadata regarding the test. The measurements were taken for 8 different CNC milling machines, between the years 2013 and 2022. The machines had measurements for each operational plane (XY, YZ, and ZX) for two different feed rates, 1000 and 4000 mm/min. The service reports were made available for each machine, and paired with the corresponding ballbar measurement. The report document contained the date of the report, a description of the severity of the fault (yellow or red), and a short description of the cause of the fault and the affected plane (or axes). A yellow marked report indicated a developing, but non-critical fault, whereas a red marking meant a need for immediate repairs. All ballbar measurements that did not have a corresponding report signify a still healthy machine. Thus, in order to predict the RUL of a machine, the measurements corresponding to the red reports were set to have a RUL value of 0. There was some discrepancy in the representation of different axis in the fault reports. Thus, in order to ensure a balanced dataset, only the measurement data and reports regarding the ZX plane was used.

However, fitting a DL model on the data at this stage would be difficult for two reasons. Primarily, the data is still sparse, meaning that many machines have only a handful of measurements between failures. Secondly, there is no standardized interval in the data, this would likely confuse any model, as an unknown amount of time will have passed between measurements. This also makes it unclear how far into the future the prediction will be expected to fall. In order to solve both problems, the data was interpolated. The up-sampling frequency was set to weekly, as to generate adequate samples for training while still maintaining interval between tests that realistically could be carried out by an operator. After discussions with domain experts, it was decided that interpolation in this manner makes sense. As we expect a fault to be evolving over time slowly moving from one measured state to another. This is similar to the procedure used to generate the well-known C-MAPSS dataset (Saxena et al., 2008), which consists of simulated run-to-failure turbofan engines. In the simulation, a fault grows over time making each time step in the data correlated with those before and after.

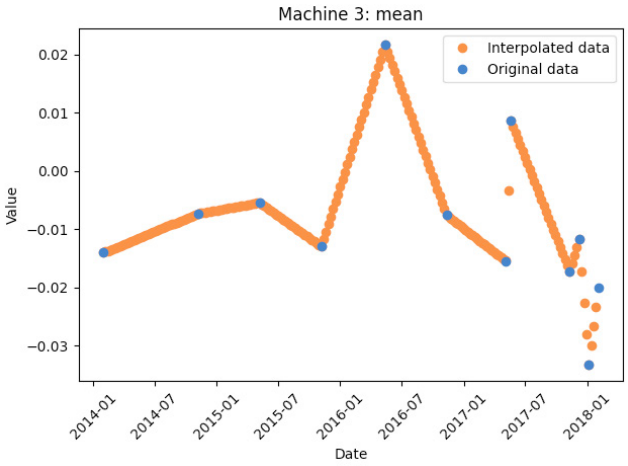

Fig. 1. Example of an interpolated feature. The blue dots represent the mean of the circularity deviations for each measurement date, generated by `tsfresh`. The yellow dots are the (weekly) interpolated values. The rightmost measurement in early 2018 is the date of machine failure.

However, the start and end points of the circularity deviation data is not guaranteed to line up between measurements. In addition, the measurements are not always of the same sample length, making interpolating between measurement points difficult. To alleviate these issues, and make the process generalizable to any time series data, the measurements were first processed by the Python package `tsfresh` (Christ et al., 2018) with the minimal parameters setting. This resulted in 10 features per measurement and direction. Out of these 10 features, nine were used, namely: *sum values, median, mean, standard deviation, variance, root mean square, maximum, absolute maximum, and minimum*. The feature *data length* was discarded, due to it not containing information regarding the state of the machine. These features were generated for both feed rates (1000 and 4000 mm/min) and directions (CW and CCW), resulting in 36 total features per measurement date.

Once generated, each feature is linearly interpolated to yield weekly measurements, the result of one such interpolated feature can be seen in Fig. 1. One question remains, however, regarding how far ahead we can expect faults to become visible in the data. According to domain experts, the earliest we can expect to detect a fault is 26 weeks before a breakdown occurs. It is then reasonable to assume that for RUL values of more than 26 weeks, the fault may not yet have developed (or be so small as not to be detectable). Thus, we create a piece-wise linear RUL function (Sateesh Babu et al., 2016) by bounding the upper RUL value to 26. In addition, in order to avoid a strong bias towards high RUL values in the dataset, each machine was limited to the last 53 weeks of measurement

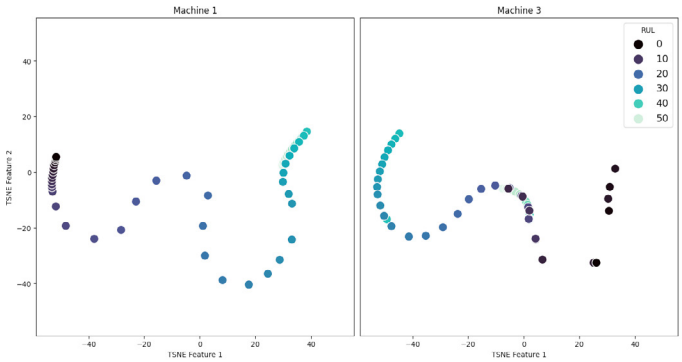

Fig. 2. t-SNE visualization: Isolated plots of the feature values of machine 1 and 3, projected on a two-dimensional plane with t-SNE. Although containing similar points, the faults occur at opposite ends of the feature space.

data (one year of runtime, in addition to the week for RUL value 0).

4. SELECTED MODEL

The DL model used in this paper is a convolutional neural network inspired by related work (see Section 2). It consists of two convolutional layers with sigmoid activation function. The kernel size is (3, 15) and (1, 8) respectively. The stride length of the kernels is (1, 3) for the first layer, and (1, 1) for the second. These are followed by a fully connected layer with 1 output neuron and a leaky ReLU activation function. Between the convolutional layers is one mean pooling layer with kernel size (2, 1). In the case of MC Dropout and the ensemble network, there is one dropout layer after each convolutional layer. The dropout layers have a dropping probability of 0.1 for the first layer, and 0.25 for the second. To reduce the risk of overfitting, the size of the model was iteratively fine-tuned to maximize predictive power at a low model parameter count.

5. EXPERIMENTS

In a first step of data exploration, the high dimensional feature data (described in Section 3) was dimensionally reduced with t-SNE (Van der Maaten and Hinton, 2008), by the scikit-learn (Pedregosa et al., 2011) Python package. From the t-SNE projection it became clear that there were at least three groups of different degradation behavior. Examples of this are shown in Fig. 2. Here, it is apparent that the two machines, while occupying a similar feature space, experience faults at opposite ends of the space, this means that fitting a single model for all machines will have trouble distinguishing high RUL values from low RUL values. For the following experiments, we focus on machines that are similar both in the type of fault experienced (given by the written reports), and close in the t-SNE projection space. Some thoughts on how to deal with this challenge, and apply a more general model is discussed in Section 7.

Similar to the related work, the data was gathered into time windows. After manual tuning, a data window size of 10 was found as a balance point between predictive power and dataset reduction. All models are evaluated using the MAE and RMSE of the mean of the prediction distribution. Likewise, the shape of the prediction distribution given by each model is compared by utilizing the CRPS and Reliability Score metrics.

For all experiments, the deep ensemble and MC Dropout models were trained with 10% of the training data set aside for validation. To prevent over-fitting, a decaying learning rate was implemented. For VI, the likelihood was modeled in Turing with a Gaussian with σ^2 set to 1, and μ yielded by the NN. For inference, NN weights were sampled from the learned distribution to yield a point prediction μ. Because of the windowed data, randomly picking samples may lead to a high overlap between training and validation sets. Thus, the validation data was selected in one coherent interval at the end of the training data.

For experiments containing only one run-to-failure interval in the training data (experiment 1 and 2.2), this was set to the high end of the RUL values. For experiments with multiple intervals (experiment 2.1) in the training set, the validation data was picked from the low-end of RUL values from one of the intervals. For an overview of which machine measurement data used in each experiment, see Table 1. The number of predictions each of the three methods gives per sample from the test data was set to 100. This means that for MC Dropout, the fully trained model was fed the same test data 100 times. For VI, the model weights were sampled 100 times to create 100 unique models. The ensemble model used 100 individually trained networks where each network contributes one prediction per test sample.

Table 1. Machine faults used per experiment

Experiment	Training Set	Testing Set
Exp. 1	Machine 1, fault 1	Machine 1, fault 2
Exp. 2.1	Machine 1, fault 1,2	Machine 2 fault 1
Exp. 2.2	Machine 3, fault 1	Machine 4, fault 2

5.1 Experiment 1

The first experiment sets out to test whether data from a machine could be used to predict a similar future fault on the same machine. Machine 1 was alone in having two recorded faults which both has a matching report and fell into the same cluster in projection space. The training set spanned a full year. However, due to long periods without measurements, the testing set contained only 18 weeks of data. After applying the time window, this resulted in a test set of nine samples (week 0 to 8).

5.2 Experiment 2

The second experiment is designed to test to what degree a model trained on one machine can be used to predict faults on another machine. This is tested for two pairs of machines. For the first pair, both failures from machine 1 is collected into a training set, where the samples from the shorter second sample interval is repeated to create two intervals of similar length. The data from the first fault

of machine 2 was identified as similar, and is withheld as a test-set. The second pair consists of training data from the only fault of machine 3 and test data from the second fault of machine 4.

6. RESULT AND DISCUSSION

The model architecture, along with the three fitting methods were implemented in the Julia programming language with the Flux (Innes, 2018) and Turing (Ge et al., 2018) packages. The resulting metrics for each model and experiment can be seen in Table 2. In the table, all metrics are the mean taken over 10 individual runs. Examples of predictions from the MC Dropout model on the test data of experiment 1, 2.1, and 2.2 can be seen in Fig. 3.

Table 2. RMSE, MAE, CRPS, and RS for all models and experiments. The unit is in weeks. Lower is better, the best result is marked in bold.

Experiment	Metric	MC Dropout	VI	Ensemble
Exp. 1	RMSE	**1.34**	2.90	1.65
	MAE	**0.99**	2.65	1.25
	CRPS	0.80	1.64	**0.79**
	RS	0.16	0.27	**0.07**
Exp. 2.1	RMSE	8.15	**7.97**	8.19
	MAE	**5.33**	5.34	5.36
	CRPS	10.57	11.31	**10.55**
	RS	**0.41**	0.47	**0.41**
Exp. 2.2	RMSE	2.98	11.85	**2.96**
	MAE	2.50	11.25	**2.32**
	CRPS	2.10	10.77	**1.60**
	RS	0.33	0.50	**0.05**

Evaluating model performance, MC Dropout and deep ensemble have better point prediction accuracy and uncertainty estimation when compared to VI. Although both the deep ensemble and MC Dropout models are close in point prediction error. The ensemble consistently performs equally well, or better, when it comes to Uncertainty Quantification (UQ). Hence, the deep ensemble may be the preferred choice in these aspects. However, as the deep ensemble is much costlier to train when compared to MC Dropout (about 100 times more time consuming), the scalability of the deep ensemble may come into question.

Picking similar machines for training and testing likely induces some bias in the test results. However, picking machines that are similar does not guarantee high performance, as can be seen in experiment 2.1. Here all models failed to find the degradation trend for machine 2 in time, even though machine 1 was similar in feature space (t-SNE plot) and in the type of fault (from the service report). A potential contributing factor to this behavior is that the dates may not be completely correlated to degradation behavior. The production load may be different from machine to machine; or it may indeed shift over time, for example during vacations. Instead of measuring RUL in weeks, as done in this paper, perhaps a more descriptive metric to use would be the number of units produced between the measurements.

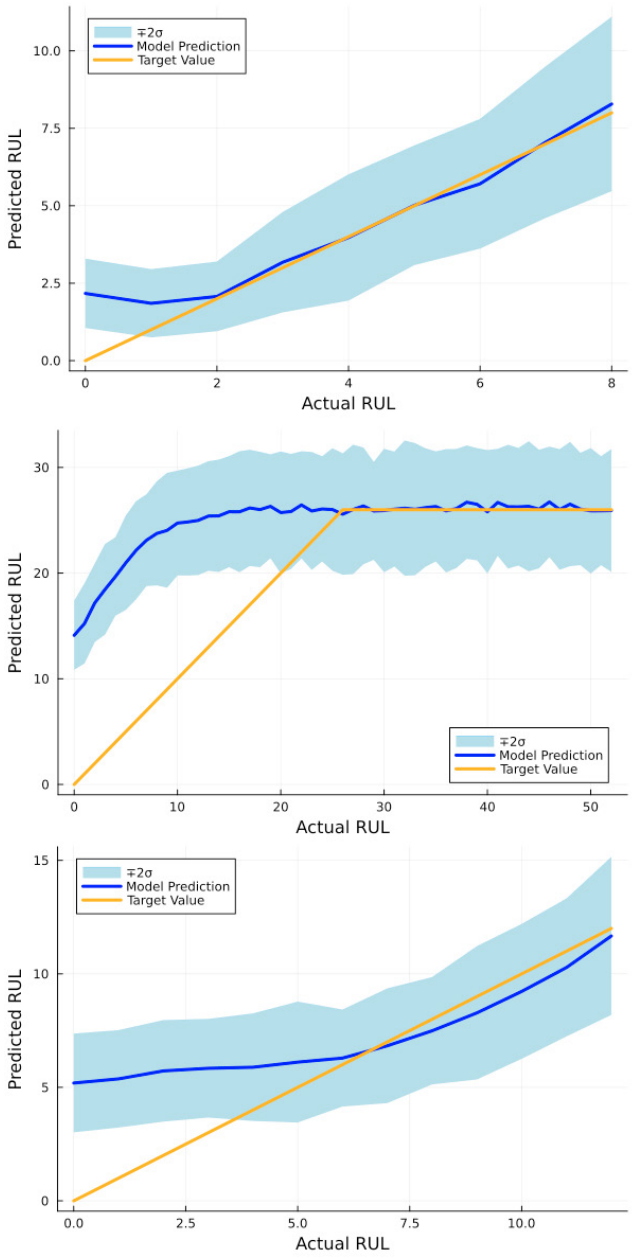

Fig. 3. Predictions of the MC Dropout model for the test data of experiment 1 (top), 2.1 (middle), and experiment 2.2 (bottom).

7. CONCLUSIONS

In this paper, we have demonstrated a novel approach for interpolating ballbar measurement data in order to train DL models. The `tsfresh` extraction procedure is domain agnostic, and can be applied to other time-series data where it can produce enough new data to enable the use of DL models. This opens up for more applications of other datasets in the future, as NNs can now be applied on datasets that have previously been prohibitively small. We also train and compare three different probabilistic NNs, providing valuable points of comparisons for future studies. Finally, we show how visualizing the interpolated data can assist in the design of DL models.

During preliminary testing, it was found that a single NN model did not perform well when trained on the total dataset of all machines. In fact, the t-SNE projection of the extracted features indicates that there are multiple different degradation behaviors within the same cluster of machines. These behaviors may be better distinguished by extracting additional `tsfresh` features. Another potential avenue of research to increase generalizability is to add a step before the final RUL prediction. Here, we can first cluster the machines according to the sub-clusters identified within the data. We can then fit a dedicated RUL predictor model to each cluster. Future measurements would first be sorted into the correct cluster, in order to have the optimal regression model applied. In an uncertainty aware system, this could also serve as way to warn the operators of a new developing machine fault that is different from previous encountered errors.

REFERENCES

Achouch, M., Dimitrova, M., Ziane, K., Sattarpanah Karganroudi, S., Dhouib, R., Ibrahim, H., and Adda, M. (2022). On predictive maintenance in industry 4.0: Overview, models, and challenges. *Applied Sciences*, 12(16), 8081.

Basora, L., Viens, A., Chao, M.A., and Olive, X. (2025). A benchmark on uncertainty quantification for deep learning prognostics. *Reliability Engineering & System Safety*, 253, 110513.

Benker, M., Furtner, L., Semm, T., and Zaeh, M.F. (2021). Utilizing uncertainty information in remaining useful life estimation via bayesian neural networks and hamiltonian monte carlo. *Journal of Manufacturing Systems*, 61, 799–807.

Blei, D.M., Kucukelbir, A., and McAuliffe, J.D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518), 859–877.

Čep, R., Malotová, Š., Kratochvíl, J., Stančeková, D., Czán, A., and Jakab, T. (2018). Diagnosis of machine tool with using renishaw ball-bar system. In *MATEC Web of Conferences*, volume 157, 01006. EDP Sciences.

Chen, C., Shi, J., Lu, N., Zhu, Z.H., and Jiang, B. (2022). Data-driven predictive maintenance strategy considering the uncertainty in remaining useful life prediction. *Neurocomputing*, 494, 79–88.

Christ, M., Braun, N., Neuffer, J., and Kempa-Liehr, A.W. (2018). Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing*, 307, 72–77.

de Pater, I. and Mitici, M. (2022). Novel metrics to evaluate probabilistic remaining useful life prognostics with applications to turbofan engines. In *PHM society European conference*, volume 7, 96–109.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 1050–1059. PMLR.

Ganaie, M.A., Hu, M., Malik, A.K., Tanveer, M., and Suganthan, P.N. (2022). Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115, 105151.

Ge, H., Xu, K., and Ghahramani, Z. (2018). Turing: a language for flexible probabilistic inference. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, 1682–1690. URL http://proceedings.mlr.press/v84/ge18b.html.

Hansen, L.K. and Salamon, P. (1990). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10), 993–1001.

Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R.R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hoffman, M.D., Gelman, A., et al. (2014). The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1), 1593–1623.

Innes, M. (2018). Flux: Elegant machine learning with julia. *Journal of Open Source Software*, 3(25), 602.

Kompa, B., Snoek, J., and Beam, A.L. (2021). Second opinion needed: communicating uncertainty in medical machine learning. *NPJ Digital Medicine*, 4(1), 4.

Lee, J., Ni, J., Djurdjanovic, D., Qiu, H., and Liao, H. (2006). Intelligent prognostics tools and e-maintenance. *Computers in industry*, 57(6), 476–489.

Nguyen, K.T., Medjaher, K., and Gogu, C. (2022). Probabilistic deep learning methodology for uncertainty quantification of remaining useful lifetime of multi-component systems. *Reliability Engineering & System Safety*, 222, 108383.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Sarker, I.H. (2021). Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN computer science*, 2(6), 420.

Sateesh Babu, G., Zhao, P., and Li, X.L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. In S.B. Navathe, W. Wu, S. Shekhar, X. Du, X.S. Wang, and H. Xiong (eds.), *Database Systems for Advanced Applications*, 214–228. Springer International Publishing, Cham.

Saxena, A., Goebel, K., Simon, D., and Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, 1–9. IEEE.

Si, X.S., Wang, W., Hu, C.H., and Zhou, D.H. (2011). Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, 213(1), 1–14.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Youness, G. and Aalah, A. (2023). An explainable artificial intelligence approach for remaining useful life prediction. *Aerospace*, 10(5), 474.

Zonta, T., Da Costa, C.A., da Rosa Righi, R., de Lima, M.J., da Trindade, E.S., and Li, G.P. (2020). Predictive maintenance in the industry 4.0: A systematic literature review. *Computers & Industrial Engineering*, 150, 106889.