Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023)
Edited by Mário P. Brito, Terje Aven, Piero Baraldi, Marko Čepin and Enrico Zio

Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-8071-1_P561-cd

Data-Driven Remaining Useful Life Estimation of Discrete Power Electronic Devices

Wilhelm Söderkvist Vermelin
RISE Research Institutes of Sweden E-mail: wilhelm.soderkvist.vermelin@ri.se

Andreas Lövberg
RISE Research Institutes of Sweden E-mail: andreas.lovberg@ri.se

Maciej Misiorny
QRTECH AB E-mail: maciej.misiorny@qrtech.se

Mattias P. Eng
RISE Research Institutes of Sweden E-mail: mattias.eng@ri.se

Klas Brinkfeldt
RISE Research Institutes of Sweden E-mail: klas.brinkfeldt@ri.se

Robust and accurate prognostics models for estimation of remaining useful life (RUL) are becoming an increasingly important aspect of research in reliability and safety in modern electronic components and systems. In this work, a data driven approach to the prognostics problem is presented. In particular, machine learning models are trained to predict the RUL of wire-bonded silicon carbide (SiC) metal–oxide–semiconductor field-effect transistors (MOSFETs) subjected to power cycling until failure. During such power cycling, ON-state voltage and various temperature measurements are continuously collected. As the data set contains full run-to-failure trajectories, the issue of estimating RUL is naturally formulated in terms of supervised learning. Three neural network architectures were trained, evaluated, and compared on the RUL problem: a temporal convolutional neural network (TCN), a long short-term memory neural network (LSTM) and a convolutional gated recurrent neural network (Conv-GRU). While the results show that all networks perform well on held out testing data if the testing samples are of similar aging acceleration as the samples in the training data set, performance on out-of-distribution data is significantly lower. To this end, we discuss potential research directions to improve model performance in such scenarios.

Keywords: electronics, prognostics and health management, remaining useful life, data-driven, machine learning, deep learning, power cycling

1. Introduction

1.1. *Background*

Modern electronic components are increasingly complex which, in turn, increases the risk of failure. In order to ensure the reliability and safety of these components, it is advantageous to be able to predict the remaining useful life (RUL) of the components. This problem is especially important in the context of safety critical applications such as aerospace, automotive, and medical devices. One key aspect of RUL prediction is the ability to collect data from the components during their operation, especially when subject to varying operating conditions. Furthermore, these data need to contain complete run-to-failure trajectories of the components in order to capture the entire degradation process and enable the use of supervised machine learning.

However, collecting such data from components being in use has historically been associated with increased risk and high costs. Interestingly, the cost of collecting relevant data via various sensors has recently decreased. Despite this, data sets from operating components, containing complete run-to-failure trajectories, are still scarce. To alleviate this issue and to drive research in this area, a data set consisting of the degradation histories of 33 wire-bonded silicon carbide (SiC) metal–oxide–semiconductor field-effect transistors (MOSFETs) subjected to power cycling until failure was recorded. The SiC MOSFETs were cycled at different loadings, i.e., varying degree of aging rate, leading to a substantial variety in degradation and life lengths.

In this work, a data-driven approach to the RUL estimation problem is presented. In particular, machine learning models are used to predict the RUL of the SiC MOSFETs. The machine learning models studied are various variants of deep neural networks (DNNs). In particular, temporal convolutional neural networks (TCNs) Lea et al. (2016), long short-term memory neural networks (LSTMs) Hochreiter and Schmidhuber (1997), and convolutional gated recurrent neural networks (Conv-GRUs) Ballas et al. (2016), are evaluated and compared on the RUL estimation task.

1.2. *Derived Features Data Set*

The data set used in the presented work is a collection of SiC MOSFETs power cycled under diverse loadings. The MOSFETs are cycled in a power cycling rig which allows for continuous monitoring of the voltages and various temperatures. The rig outputs data in a structured text-format which are parsed to obtain time series of the collected measurement quantities. This process is shown in Figure 1.

Devices under test (DUTs) are cycled until either of the set rig failure conditions are met:

- measured ON-state voltage is higher than 8 V, or
- recorded MOSFET casing temperature is higher than 100 °C.

The settings for experiment rounds one through five is shown in Table 1.

After the data have been parsed in a useful format, key characteristics for each cycle are extracted. As a result, one obtains a final data set consisting of 21 features recorded per cycle, namely: The *cycle count* is the number of cycles the DUT has been cycled and the *consumed life* is the fractional life left until failure (bounded in the interval $[0, 1]$). *End voltage* and *end resistance* correspond to the voltage and resistance measured at the end of the cycle, respectively. Note that

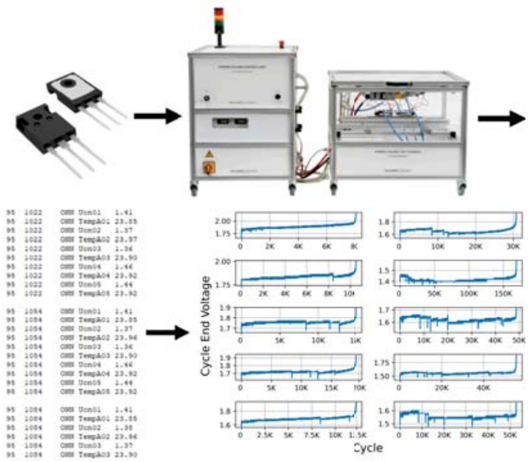

Fig. 1. The data generation process, from power cycled MOSFETs to time series data, making data amenable for machine learning modeling.

Table 1. SiC MOSFETs data set, indexed by experiment number.

Exp. no. (#)	Current (A)	DUTs (#)	ON (s)	OFF (s)
1	25 A	5	10	10
2	23 A	10	10	10
3	24 A	10	10	10
4	28 A	8	10	10
5	25 A	10	15	10

"min", "max" and "mean" prefixes translate to the lowest, highest and average recorded measurement of the corresponding feature within the cycle time frame, respectively. Furthermore, *end resistances from max leg* and *mean block temperatures* stand for end resistances one should expect for a system in the "healthy" (not degraded) state, and they have been estimated using two different methods based on specific temperatures as indicated. *Residual end resistances* represent then the accumulated damaged obtained by subtracting the "healthy" resistances from the actual resistances. Finally, the "clean" qualifier means that the data have been additionally subjected to some data cleaning procedure. A summary of the derived features is shown in Table 2. The data set of derived features is the basis for the RUL estimation

Table 2. Derived features data set.

Feature	Unit
Cycles	unitless
Consumed life	unitless
End voltage	V
End resistance	mΩ
Min leg temperature	$^\circ$C
Max leg temperature	$^\circ$C
End resistance from max leg temperature	mΩ
Residual end resistance from max leg temperature	mΩ
Cleaned residual end resistance from max leg temperature	mΩ
Min temperature	$^\circ$C
Max temperature	$^\circ$C
Min block temperature 1	$^\circ$C
Max block temperature 1	$^\circ$C
Min block temperature 2	oC
Max block temperature 2	$^\circ$C
Min water inlet temperature	$^\circ$C
Max water inlet temperature	$^\circ$C
Min water outlet temperature	$^\circ$C
Max water outlet temperature	$^\circ$C
Mean block temperature	$^\circ$C
End resistance from mean block temperature	mΩ
Residual end resistance from mean block temperature	mΩ
Cleaned residual end resistance from mean block temperature	mΩ

task where RUL is related to consumed life (CL) as follows: $\mathrm{RUL} = 1 - \mathrm{CL}$.

2. Methodology

2.1. *Data Preprocessing*

Although a considerable amount of effort was devoted to create a set of derived features, a data set developed for RUL estimation modeling, some data preprocessing was needed to make the data suitable for machine learning. The performed data preprocessing is consistent with common practice in machine learning, namely, it involves splitting data into training and holdout data and standardizing data to make features equally scaled. As a first step, data are split into training, validation, and testing sets. This allows for training the model on a subset of the data, tuning its hyperparameters on another subset, and finally evaluating its performance on a holdout set. The splitting of the data into the training, validation, and testing sets was carried out in such a way that MOSFETs from experiments with a higher rate of aging were used in training and validation sets while MOSFETs from the least accelerated aging experiment (23 A, experiment 2, cf. Table 1) were used for the held out testing data set. This approach is used to enable a more accurate estimation of the models' performance for operating conditions outside the training data distribution.

Next, data are *standardized* based on the training data. This step involves subtracting the mean and dividing by the standard deviation,

$$\tilde{\mathbf{X}} = \frac{\mathbf{X} - \bar{\mathbf{X}}}{\mathbf{s}} \tag{1}$$

where

$$\bar{X}^{(p)} = \frac{1}{N} \sum_{i=1}^{N} X_i^{(p)} \tag{2}$$

$$\mathbf{s} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(X_i^{(p)} - \bar{X}^{(p)} \right)^2} \tag{3}$$

and $p = 1, \ldots, P$, with P being the number of features in the data set. This ensures that all features are weighted equally and the model can learn meaningful patterns from the data. Here, it is important to standardize validation and test data based on the feature averages and standard deviations based on training data to avoid data leakage.

2.2. *Remaining Useful Life Estimation*

The remaining useful life estimation problem can be formulated as a supervised learning problem since the data consists of complete run-to-failure trajectories. It means that a mapping f from a collection of sensor time series data, $\mathcal{D}$, to RUL, y, is learned:

$$f : \mathcal{D} \mapsto y. \tag{4}$$

In this case, $\mathcal{D}$ is the set of features per cycle as given in Table 2 and y is the RUL, for each MOSFET.

2.3. *Model Architectures*

The models chosen for RUL estimation represent different flavors of deep neural networks, designed for time series data. Essentially, artificial neural networks consists of layers of artificial neurons connected by so called weights. In a Feed Forward Neural Network (FFNN) Goodfellow et al. (2016) the forward pass from one layer to the next is defined by

$$\mathbf{h}^{(l)} = \sigma^{(l)} \left(\mathbf{W}^{(l)\top} \mathbf{h}^{(l-1)} + \mathbf{b}^{(l)} \right), \qquad (5)$$

for $l = 2 \dots L$, where $\mathbf{h}^{(l)}$ is the input to the layer l (with $l = 1$ corresponding to the input layer such that $\mathbf{h}^{(1)} = \mathbf{x}$, where $\mathbf{x}$ represent the input data). $\mathbf{W}^{(l)}$ is the weight matrix containing the weights connecting layer $l - 1$ and l and $\mathbf{b}^{(l)}$ is what is called the bias. L stands for the number of layers in the neural network and σ represents a non-linear transformation referred to as the activation function. In general, different layers may have different activation functions, hence the subscript in Equation (5). A common choice for activation function is the sigmoid activation function

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \qquad (6)$$

or the rectified linear unit (ReLU)

$$\sigma(x) = \begin{cases} 0 \text{ for } x \leq 0, \\ x \text{ for } x > 0. \end{cases} \qquad (7)$$

2.3.1. *Temporal Convolutional Neural Network*

The Temporal Convolutional Neural Network (TCN) is a type of neural network architecture that is designed to process sequential data, such as time series or natural language sentences Lea et al. (2016). It is based on the Convolutional Neural Network (CNN) architecture LeCun et al. (2015), which is commonly used for image recognition tasks. The TCN architecture consists of a series of convolutional layers, each of which applies a set of filters to the input sequence. The filters are designed to capture different patterns in the data, such as short-term dependencies or long-term trends. The output of each convolutional layer is then passed through a non-linear activation function, such as a ReLU (cf. Equation (7)), to introduce non-linearity into the model. One of the key features of the TCN architecture is the use of dilated convolutions, which allow the network to capture patterns at different time scales. A dilated convolution is a convolutional operation where the filter is applied to the input sequence with gaps between the values. This allows the filter to capture patterns that are spread out over a larger time window, without increasing the number of parameters in the model. Another important aspect of the TCN architecture is the use of residual connections, which help to mitigate the problem of vanishing gradients. A residual connection is a shortcut connection that allows the output of one layer to be added to the input of a later layer. This helps to ensure that the gradients can flow through the network more easily, which can improve the training process. TCNs have been used in RUL prediction previously, e.g., for predicting lithium ion battery failure Zhou et al. (2020).

2.3.2. *Long Short-Term Memory Neural Network*

The Long Short-Term Memory (LSTM) neural network Hochreiter and Schmidhuber (1997) is a type of Recurrent Neural Network (RNN) that is designed to handle the problem of vanishing gradients in traditional RNNs. LSTM networks are capable of learning long-term dependencies in sequential data by selectively retaining or forgetting information over time. The architecture of an LSTM network consists of a series of memory cells that are connected to each other through gates. These gates control the flow of information into and out of the memory cells, allowing the network to selectively retain or discard information based on its relevance to the current task. Each memory cell in an LSTM network has three main components: an input gate, a forget gate, and an output gate. The input gate controls the flow of new information into the cell, the forget gate controls the retention or discarding of old information, and the output gate controls the flow of information out of the cell. LSTM networks have been successfully applied to a wide range of tasks, including speech recognition, natural language processing, and image captioning. Such networks are particularly useful in tasks that involve long-term dependencies, such as, predicting the next word in a sentence or generating a sequence of musical notes. LSTMs have been shown to perform well in RUL prediction, e.g., turbofan engines, lithium ion batteries, and other applications Zhang et al. (2018); Zheng et al. (2017).

2.3.3. *Convolutional Gated Recurrent Unit*

The Convolutional Gated Recurrent Unit (Conv-GRU) is a combination of two model architectures; an RNN and a CNN. A GRU is a type of RNN that is designed to process sequential data, such as time series or natural language sentences Cho et al. (2014). It was introduced as an improvement over the traditional RNN architecture. The GRU architecture consists of a series of recurrent units, each of which has a gating mechanism that controls the flow of information. The gating mechanism consists of two gates: an update gate and a reset gate. The update gate determines how much of the previous state should be retained, while the reset gate determines how much of the new input should be incorporated into the state. The update gate is computed using a sigmoid function, which outputs a value in the range between 0 and 1, defined in Equation (6). Here, 0 means that the previous state is completely ignored, while a 1 means that the previous state is completely retained. The reset gate is also computed using a sigmoid function, but it is applied element-wise to the input and the previous state. It allows the network to selectively forget or remember certain parts of the input sequence.

The Conv-GRU consists of multiple layers of GRU cells, each of which is connected to a convolutional layer. The convolutional layer processes the input data at each time step, and the output is fed into the GRU cell. The GRU cell then updates its hidden state based on the input and the previous hidden state, and produces an output that is passed to the next layer Ballas et al. (2016).

2.4. *Model Training*

The models under consideration are trained for one hundred epochs (an epoch corresponds to one full pass of the training data set), with an initial learning rate of 0.001 and their performance is evaluated using Mean-Squared-Error (MSE) loss:

$$\ell(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 . \qquad (8)$$

The models are optimized through gradient descent on the loss function using the AdamW optimizer Kingma and Ba (2015); Loshchilov and Hutter (2019). The neural networks are implemented in Python Van Rossum and Drake (2009), using the deep learning framework PyTorch Paszke et al. (2019) and a wrapper built on top of PyTorch called PyTorch Lightning Falcon and team (2019).

3. Results

The performance of the models is quantified in terms of MSE, see Equation (8). In Table 3, the MSE on the validation and test data sets are presented. In Figure 2, 3, and 4 the RUL evaluation plots for TCN, Conv-GRU and LSTM are shown, respectively. The evaluation plot presents the RUL (scaled in the domain $[0, 1]$) as a function of the number of cycles. The blue points forming a straight line depict the actual RUL (ground truth) whereas the orange points represent predicted RUL. These plots give insight into whether the model tends to overestimate or underestimate RUL.

Table 3. Model performance in terms of MSE. The most performant model scores are highlighted with bold text.

	MSE	
Model	Validation	Test
TCN	0.052	0.033
ConvGRU	**0.025**	**0.032**
LSTM	0.036	0.0607

4. Discussion and Conclusions

In this paper an estimation of the Remaining Useful Life (RUL) based on deep learning was presented for silicon carbide metal-oxide field effect

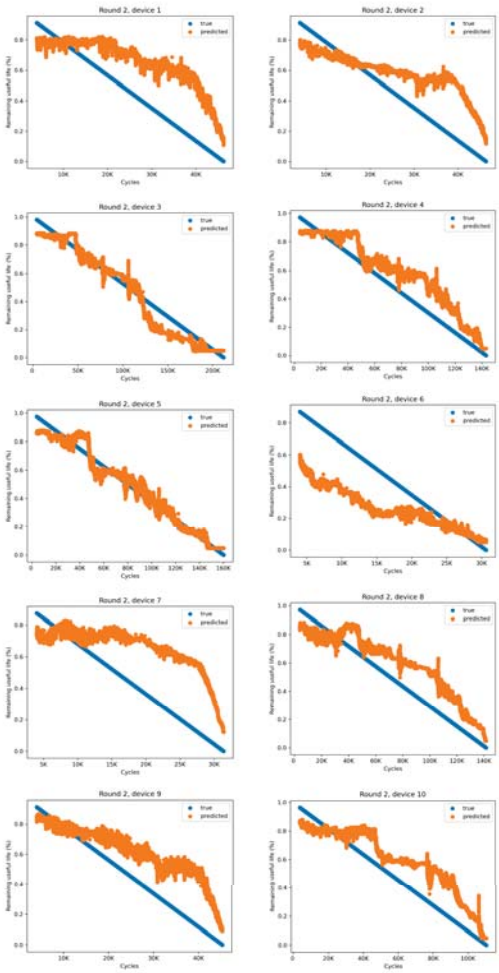

Fig. 2. TCN model RUL estimation plots for test data samples.

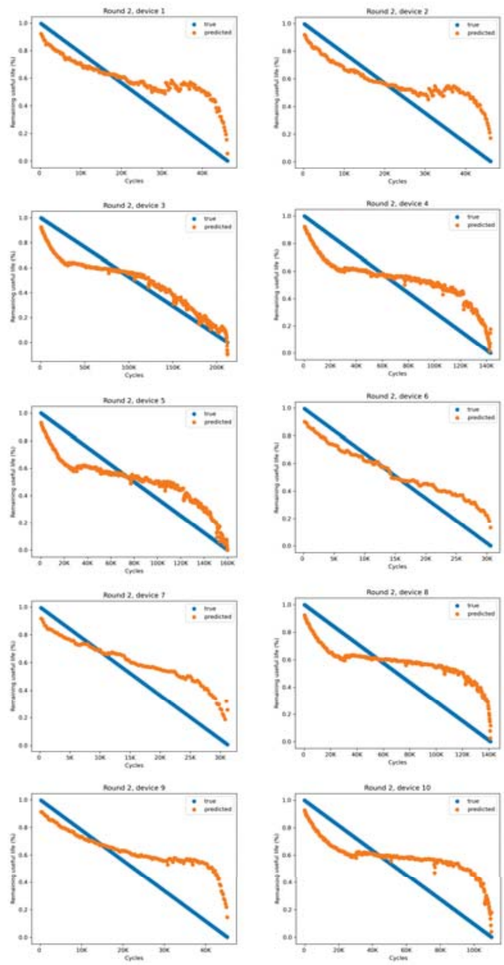

Fig. 3. Conv-GRU model RUL estimation plots for test data samples.

transistors (SiC MOSFETs). The data set obtained by accelerated aging of MOSFETs is a novel data set for developing prognostics models for MOSFETs. Three different deep learning model architectures were compared and evaluated. It was shown in Table 3 that the models perform well on RUL estimation task, particularly for devices that were degraded at a similar pace as those in the training data set. The Conv-GRU model and the TCN model performed best on test set data (data coming from a more gentle aging experiment round) with the Conv-GRU model slightly outperforming the TCN model. The Conv-GRU model has good performance while also being parameter efficient, it has more than 100 times fewer parameters than the TCN model that was trained.

The performance of the models, however, deteriorates when making predictions on devices subjected to more benign cycling conditions, as can be observed in Figures 2, 3, and 4, leading to an overestimation of RUL close to end-of-life. This behavior is expected, as machine learning models struggle on out-of-distribution data. To make the models useful for real-world applications (for instance, automotive), this gap needs to be bridged. A promising direction for this research

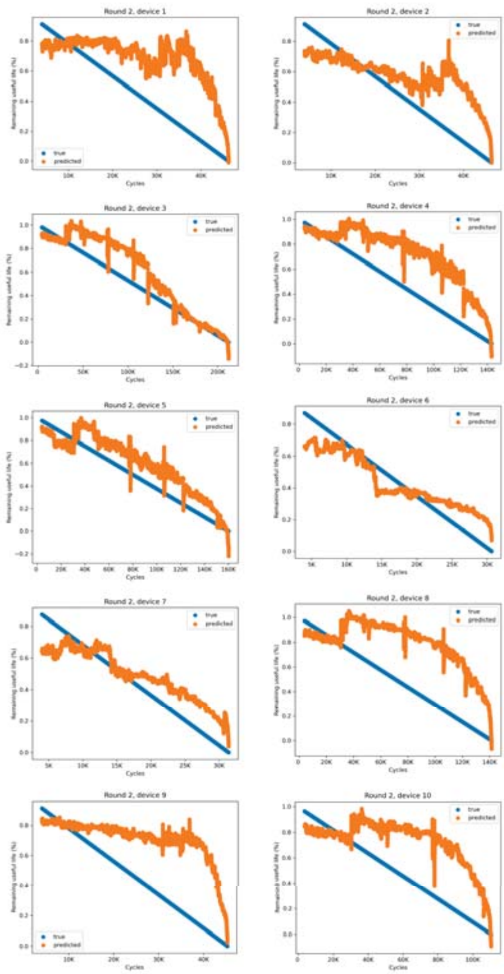

Fig. 4. LSTM model RUL estimation plots for test data samples.

is the use of Physics-Informed Machine Learning (PIML) Xu et al. (2023). PIML aims at combining knowledge of the physics of failure with data driven models in order to enhance predictive performance in situations when a sufficient amount of data is not available, and to make prognostics models more robust and reliable.

Acknowledgement

This research is conducted within the iRel4.0 Intelligent Reliability project, which is funded by Horizon 2020 Electronics Components for European Leadership Joint Undertaking Innovation Action (H2020-ECSEL-JU-IA). This work is also funded by the Swedish innovation agency Vinnova, through co-funding of H2020-ECSEL-JU-IA. The authors would also like to thank Dr. Martin Haller at RISE for insightful comments that helped improving this paper.

References

Ballas, N., L. Yao, C. Pal, and A. C. Courville (2016). Delving deeper into convolutional networks for learning video representations. In Y. Bengio and Y. LeCun (Eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Cho, K., B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio (2014, October). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1724–1734. Association for Computational Linguistics.

Falcon, W. and T. P. L. team (2019, 3). Pytorch lightning.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Hochreiter, S. and J. Schmidhuber (1997). Long short-term memory. *Neural computation 9*(8), 1735–1780.

Kingma, D. P. and J. Ba (2015). Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lea, C., R. Vidal, A. Reiter, and G. D. Hager (2016). Temporal convolutional networks: A unified approach to action segmentation. In G. Hua and H. Jégou (Eds.), *Computer Vision – ECCV 2016 Workshops*, Cham, pp. 47–54. Springer International Publishing.

LeCun, Y., Y. Bengio, and G. Hinton (2015). Deep learning. *Nature 521*(7553), 436–444.

Loshchilov, I. and F. Hutter (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Paszke, A., S. Gross, F. Massa, A. Lerer,

J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc.

Van Rossum, G. and F. L. Drake (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Xu, Y., S. Kohtz, J. Boakye, P. Gardoni, and P. Wang (2023). Physics-informed machine learning for reliability and systems safety applications: State of the art and challenges. *Reliability Engineering & System Safety 230*, 108900.

Zhang, Y., R. Xiong, H. He, and M. G. Pecht (2018). Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Vehicular Technology 67*(7), 5695–5705.

Zheng, S., K. Ristovski, A. Farahat, and C. Gupta (2017). Long short-term memory network for remaining useful life estimation. In *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pp. 88–95.

Zhou, D., Z. Li, J. Zhu, H. Zhang, and L. Hou (2020). State of health monitoring and remaining useful life prediction of lithium-ion batteries based on temporal convolutional network. *IEEE Access 8*, 53307 – 53320. Cited by: 67; All Open Access, Gold Open Access.