Probabilistic and Physics-Informed Machine Learning for Predictive Maintenance with Time Series Data

Phan-Anh Vu[1*], Emanuel Aldea[2], Mounira Bouarroudj[3], Sylvie Le Hégarat-Mascle[2]
[1] Grenoble Alpes University, CEA LIST, CNRS, Grenoble INP, GIPSA-Lab, Grenoble 38000, France
[2] Paris-Saclay University, CNRS, SATIE Laboratory, Gif-sur-Yvette 91190, France
[3] Paris EST Créteil University, CNRS, SATIE Laboratory, Gif-sur-Yvette 91190, France

Abstract

Physics-informed neural networks are capable of learning from both observation data and the underlying physical laws. Meanwhile, their implementation in real application settings requires additional considerations related to multi-objective optimization of variables with vastly different scales. Besides, many applications benefit from having well-calibrated uncertainty estimate along with the prediction. In this study, we examine physics-informed neural network for a predictive maintenance application with times series data, using a physical fatigue crack propagation model from mechanical engineering. Our goal is to attain good predictive performance, while at the same time producing correct uncertainty intervals and limiting computation cost. Moreover, we also consider as baselines some established uncertainty quantification techniques in deep learning, and we provide a detailed quantitative assessment of their calibration.

1. Introduction

Nowadays, the power electronic sector is the strongest growth market in transport applications. Semiconductor-based power switches are rapidly replacing conventional electromechanical relays in most of the main vehicle functions as well as in comfort, safety and communication applications. An additional growing segment is represented by the market for alternative propulsion technologies, in line with the new regulations for road transport aimed at the reduction of greenhouse gas emission level (30% lower by 2030 in the EU), which results in the prohibition of the marketing of combustion engine cars from 2035 in Europe. This trend is seen nowadays with more and more focus on electric vehicle concepts. The rise in electronic technology in automotive inevitably creates new demands in terms of low costs, operation under extreme environmental conditions (temperature, humidity, vibration, etc.), greater system power density, increasing miniaturization and ensure high levels of reliability. This requires an in-depth knowledge of the possible evolution of electronic components as a function of time and operating conditions. Consequently, development of reliable Remaining Useful Lifetime (RUL) models is of primary importance.

In power electronic devices, lifetime is usually obtained by subjecting them to accelerated ageing tests. The results of these tests are then extrapolated to normal operating conditions [1]. However, these empirical models do not accurately describe the local degradation mechanisms leading to the degradation of the device under study. There is therefore a significant gap between the prediction by this approach and the observation made in the real conditions. Indeed, these models can only deal with one failure mechanism at a time, known a priori. To address these limitations, the second approach is based on a detailed analysis of local failure mechanisms thanks to a detailed understanding of local damage physics. It allows to acquire a precise knowledge of the causes generating the beginning of degradation and the evolution of this degradation leading to the failure. This approach is based on the analysis of physical variables or quantities such as stress, strain or energy leading to failure. These variables are usually obtained by multi-scale finite element simulations with multi-physics coupling between electrical, thermal and mechanical [2], [3], [4]. However, finite element simulations are time-consuming and sensitive to mesh size and material properties which are often not known. This last point represents a considerable limitation for physics-based models because the lack of knowledge of the exact behavior laws of materials used in power electronics leads to an incorrect evaluation of their lifetime. Model reduction methods can be considered to simplify the parametric analysis [5], but they do not avoid the inaccuracies related to the biases of the physical model used. Physics-based analytical models are also proposed in literature for the online estimation of remaining useful lifetime. They utilize a degradation model that predicts the future state based on input that describes the current system state and the expected load levels on the system [6], [7]. In this paper we propose to use a recently proposed mixed approach using both experimental data

*This work was conducted while the first author was doing an internship at SATIE Laboratory

and physical models based on stochastic methods and deep learning techniques (Physics-informed neural networks). The objective is to increase the prediction capacity while being economic in terms of computing time.

2. Problem statement

In an IGBT power module subjected to ageing under power cycling wire-bond lift-off is a dominant degradation mode [8], [9]. Monitoring of the on-state voltage VCE continuously during aging reflect the state of crack evolution in the contact wire bond-metallization [10], [2]. As illustration, evolution of VCE until reaching the failure criteria of the IGBT modules (corresponding to 5% increase in VCE) is depicted on Figure 1. The analysis by microsection reveals that cracks appear at the interfaces and begin to propagate inside the wire bonds until lift off. The idea explored in this paper is to use the Physics-informed neural networks (PINN) to predict the evolution of cracks propagation as a function of number of cycles using VCE measurements as input data.

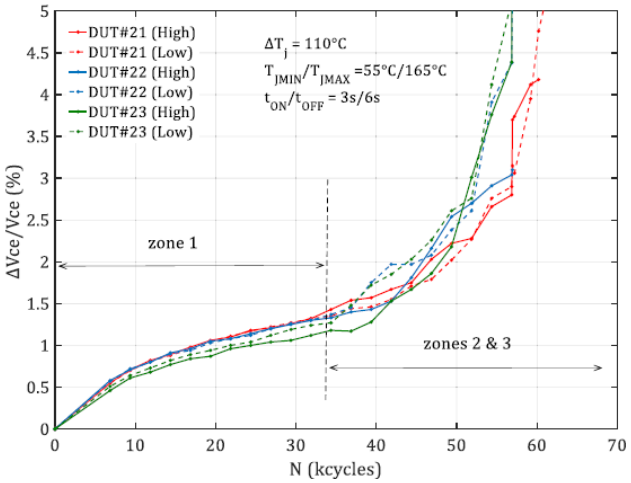


Figure 1: Evolution of collector-emitter voltage VCE for $\Delta T_j = 110°C$, $T_{jmin}0 = 55°C$ and $t_{ON}/t_{OFF} = 3s/6s$ [2].

3. Physics-informed machine learning

A. Problem formulation

With Physics-Informed Neural Network (PINN) [11], [12], the goal is to leverage information from both observation data and the underlying physical law. An observation dataset contains *n_obs* measurements, indexed by i, for input x and output y: $D = \{x_i, y_i\}; i = 1...n\_obs$. We assume there is a function h mapping input x to true output y: $y = h(x)$.

The output y need to satisfy some constraints following some physical law. Usually, the physical law involves a function of differential operator of order 1 or greater:

$$f_\nabla(y) = f\left(\nabla_x y, \nabla_x^2 y, ..., \nabla_x^d y\right) = \Phi(y) \tag{1}$$

$\nabla_x^d y$ is the derivative of order d. $\Phi(y)$ is a function defined by the physical laws. We introduce a shorthand notation for the differential part: $f_\nabla(y) = f\left(\nabla_x y, \nabla_x^2 y, ..., \nabla_x^d y\right)$.

The model is an approximator to the true function $\hat{h}$, parameterized by w, which ingests input x and produces prediction $\hat{y}$:

$$\hat{y} = \hat{h}(x; w) \tag{2}$$

The predictions should match the observations. Thus we want to minimize the difference between these two quantities, which is our observation loss. We commonly measure the squared difference, or Mean Squared Error (MSE):

$$L_{obs} = \sum_{i=1}^{n\_obs} (\hat{y}_i - y_i)^2 \tag{3}$$

On the other hand, the predictions also need to satisfy the physical law. The physical law is defined for the true output y, but at test time we only have prediction $\hat{y}$. Therefore, we enforce the physics constraint on the prediction, which is an approximation to the true target. Analogous to the observation loss, we aim to minimize the physics loss:

$$L_{phy} = \sum_{i=1}^{n\_phy} (f_\nabla(\hat{y}_i) - \Phi(\hat{y}_i))^2 \tag{4}$$

These 2 losses lead us to a multi-objective optimization problem.

B. Addressing the multi objective optimization

Multiple objectives can be combined by a simple scalarization. In a basic static scalarization scheme, we consider a weighted linear combination of the two terms above:

$$L = \lambda_{obs} L_{obs} + \lambda_{phy} L_{phy} \tag{5}$$

Finding the right value for the coefficients is critical, but it may be difficult in practice, depending on the specificity of the considered application. In a favorable situation, λ_{obs} and λ_{phy} may be optimized successfully along with other learning hyper-parameters during training. However, in some cases there are no adequate scalar coefficients leading to acceptable performance, due to some specific numerical behavior such as the physical law $\Phi(y)$ imposing a strong derivative on the model. As a consequence, a small deviation leads to a large penalty, adding thus significant instability to the learning process. Among the potential solutions to this problem, we consider here Magnitude Normalization [13], which is a heuristic to dynamically and automatically balance these two terms. The magnitude of the true target is used as coefficient for

each term in the loss summation:

$$\lambda_{obs} = \frac{1}{\sum_{i=1}^{n\_obs} y_i^2} \tag{6}$$

$$\lambda_{phy} = \frac{1}{\sum_{i=1}^{n\_phy} (|f_\nabla(\hat{y}_i)| + |\Phi(\hat{y}_i)|)^2} \tag{7}$$

$$L = \frac{L_{obs}}{\sum_{i=1}^{n\_obs} y_i^2} + \frac{L_{phy}}{\sum_{i=1}^{n\_phy} (|f_\nabla(\hat{y}_i)| + |\Phi(\hat{y}_i)|)^2} \tag{8}$$

$$= \frac{\sum_{i=1}^{n\_obs} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{n\_obs} y_i^2} + \frac{\sum_{i=1}^{n\_phy} (f_\nabla(\hat{y}_i) - \Phi(\hat{y}_i))^2}{\sum_{i=1}^{n\_phy} (|f_\nabla(\hat{y}_i)| + |\Phi(\hat{y}_i)|)^2} \tag{9}$$

A closely related method is Inverse Dirichlet Weighting [14], which uses the gradient variance to balance the terms in the loss. In our experiments, we adopt Magnitude Normalization, because it produces comparable result to Inverse Dirichlet Weighting, while requiring less computation.

C. The considered physical constraint

We focus on the application of predictive maintenance with time series data. More specifically, we try to predict the fatigue crack growth of a power converter component. The component wears down with loading cycles during usage. We seek to model the crack length as a function of the number of loading cycle.

Paris' law [15] is a common choice to model fatigue crack propagation in mechanic. Here, a is crack length, N is number of cycle, C and m are material properties, and $\Delta\sigma$ is the stress intensity.

$$\frac{da}{dN} = C(\Delta\sigma\sqrt{\pi a})^m \tag{10}$$

Matching the general notation used in previous sections, we have this correspondence for the input, output, differential function and physical law:

$$x = N; y = a; f_\nabla(y) = \frac{da}{dN}; \Phi(y) = C(\Delta\sigma\sqrt{\pi a})^m$$

By integrating both side with initial crack length a_0, we can find an analytical form of the crack length as a function of loading cycle:

$$a(N) = \left[NC\left(1 - \frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}} \right]^{\frac{2}{2-m}} \tag{11}$$

Observation datapoints are generated with Equation 11, by choosing regular samples of number of cycle N, and adding noise to the calculated crack length $a(N)$. The training procedure will use Equation 10 as the physics constraint. The parameters of Paris' law are fixed at: $C = exp(-23), m = 4, \Delta\sigma = 75, a_0 = 0.01$, following [16]. The experimental section contains multiple illustrations of data generating for training and testing following the procedure above.

4. Uncertainty quantification

In many applications, the ability to provide calibrated uncertainty measure along with prediction is invaluable. We briefly review a quantitative metric to evaluate uncertainty calibration, along with some established techniques to produce prediction interval.

A. Evaluate uncertainty calibration for regression

For a regression problem, we typically assume a Gaussian distribution over the target. Thus, the predictive distribution can be described with two parameters: mean and variance. A well-calibrated predictive distribution should match the true distribution of the target. For example, a 95% confidence interval around the predictive mean should contain the true target value 95% of the time. We can construct a calibration metric by iterating over confidence levels from 0 to 1, and count the proportion of true target lying inside the confidence interval at this level (illustrated in Figure 2, suggested by [17], implemented by [18]). A perfect calibration corresponds to the diagonal line. The miscalibration area between the two lines is a measure of calibration error.

Meanwhile, a model can always reduce the miscalibration area by inflating the prediction interval. A useful prediction interval should be as small as possible. Therefore, we also look for the smallest measure of sharpness, or the average width of the prediction interval.

B. Monte Carlo Dropout (MCDropout)

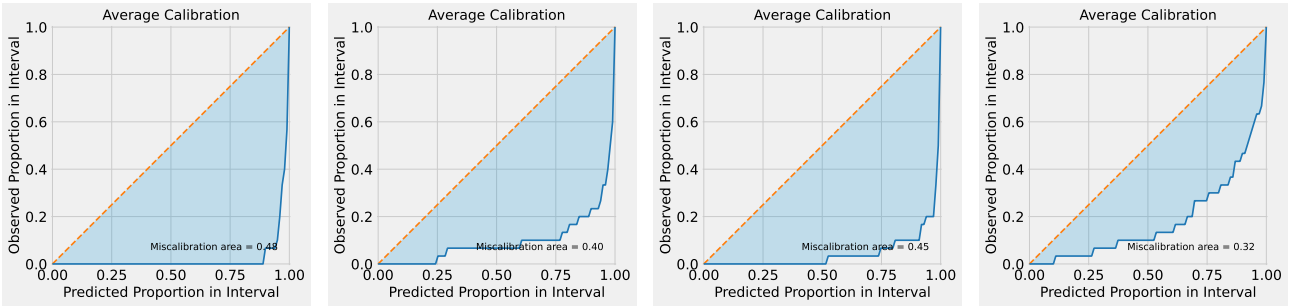
Activating dropout at test time [19] is a simple technique to approximate the posterior distribution of model parameters with samples. The main advantages of this method are ease of implementation and speed. To obtain a more consistent prediction interval, we use the same binary mask for each forward pass through the whole dataset.

C. Deep Ensemble

With ensemble methods, uncertainty quantification is based on predictive variance among individual models in the collection. Each individual model can also be considered as a sample from the parameter distribution. For regression problems, the Deep Ensemble authors [20] suggest an augmented version of the individual model. Under the assumption that the observation output follows a Gaussian distribution, the model will produce a predictive mean $\mu(x)$ and predictive variance $\sigma^2(x)$ for each input. The training objective is to minimize the negative log likelihood of the observation:

$$-\log p(y|x) = -\log\ \mathcal{N}(y|\mu(x), \sigma^2(x)) \tag{12}$$

$$= \frac{1}{2}\log 2\pi + \log\sigma(x) + \frac{(y-\mu(x))^2}{2\sigma^2(x)} \tag{13}$$



(a) MCDropout (b) MCDropout same mask (c) Bayes by backprop (d) Deep ensemble

Figure 2: Miscalibration plot for 100 samples

By setting to 0 the derivative with respect to $\sigma(x)$, we obtain the optimum:

$$\frac{d}{d\ \sigma(x)}(-\log p(y|x)) = 0$$
$$\implies \frac{1}{\sigma(x)} - (y-\mu(x))^2 \frac{1}{\sigma^3(x)} = 0$$
$$\implies \sigma^2(x) = (y-\mu(x))^2$$

As a noteworthy remark, this objective encourages the model to correctly estimate its own error. Deep ensemble is quite simple to implement. The main challenge is to ensure the diversity of the member models, which is all the more difficult for small model and small dataset.

By averaging the predictions from individual models under the Gaussian likelihood assumption, we obtain a Gaussian mixture ensemble with uniform contribution. Here, M is the number of members in the ensemble.

$$\mu(x) = \frac{1}{M}\sum_{i=1}^{M}\mu_i(x)$$
$$\sigma^2(x) = \frac{1}{M}\sum_{i=1}^{M}[\sigma_i^2(x)+\mu_i^2(x)] - \mu^2(x)$$

D. Bayes by Backprop

Bayesian Neural Network considers model parameters as random variables instead of a single value. Bayes by backprop [21] characterizes each model parameter w by a Gaussian distribution $w \sim \mathcal{N}(\mu, \sigma^2)$. Each weight and bias now has an associated mean and variance, instead of having a single value. The goal of Bayesian inference is to infer the posterior distribution over the model parameters given the data, using the likelihood and the prior distribution of the parameters.

$$p(w|D) = \frac{p(D|w)p(w)}{\int p(D|w)p(w)dw} \tag{14}$$

We often choose simple distributions such as Gaussian or uniform for the parameter prior $p(w)$. For regression problem, the likelihood $p(D|w)$ is usually a Gaussian centered on the prediction $\hat{y}$ with fixed likelihood variance $\sigma_l^2 : p(D|w) = \mathcal{N}(\hat{y}|y, \sigma_l^2)$.

Due to intractability of the normalization in the denominator, we have to approximate the true posterior $p(w|D)$ with a simpler variational distribution $q(w;\theta)$. For this specific model, the variational parameters are the means and variances of all the weights in our model: $\theta = \{\mu, \sigma^2\}$. To find a good approximation, we minimize the Kullback-Leibler divergence between the variational approximation and the true posterior distribution.

$$\theta^* = \arg\min_\theta\ KL[q(w;\theta)||p(w|D)] \tag{15}$$
$$= \arg\min_\theta \int q(w;\theta)\ \log\frac{q(w;\theta)p(D)}{p(w;D)}dw \tag{16}$$
$$= \arg\min_\theta \int q(w;\theta)\ \log\frac{q(w;\theta)}{p(D|w)p(w)}dw \tag{17}$$
$$= \arg\min_\theta \int q(w;\theta)\ \log\frac{q(w;\theta)}{p(w)}dw$$
$$- \int q(w;\theta)\ \log p(D|w)dw \tag{18}$$
$$= \arg\min_\theta\ KL[q(w;\theta)||p(w)] - \mathbb{E}_{q(w;\theta)}[\log p(D|w)] \tag{19}$$

These 2 terms, log likelihood and divergence, may need appropriate scaling. The β-VAE authors [22] suggest adding a coefficient β in front of the divergence term. This is the objective to train Bayes by backprop model. Finding the right choice of coefficient β requires experimenting with different values.

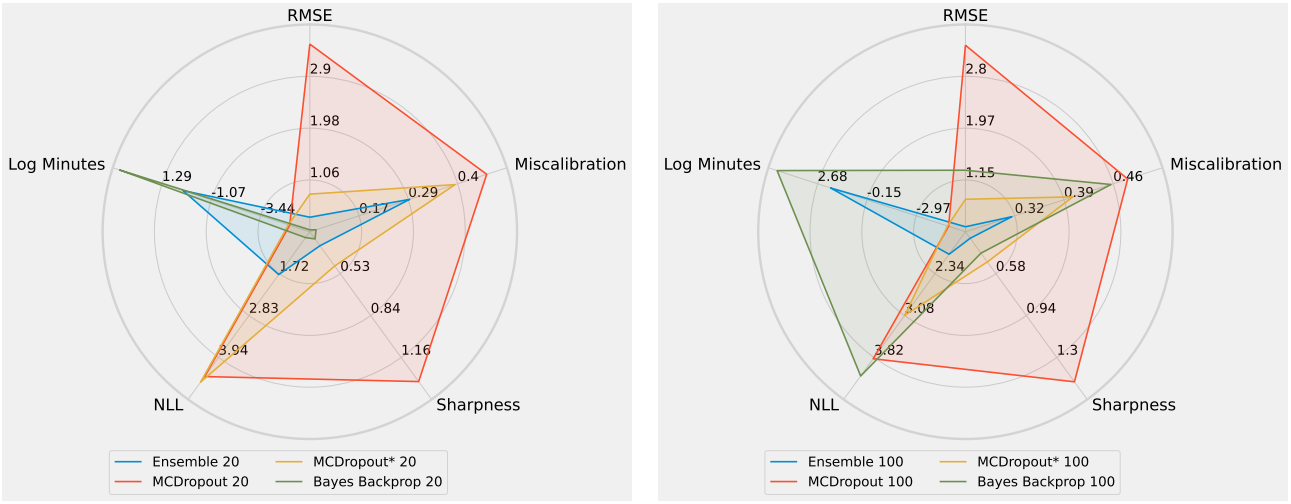
$$\theta^* = \arg\min_\theta\ \beta KL[q(w;\theta)||p(w)] - \mathbb{E}_{q(w;\theta)}[\log p(D|w)] \tag{20}$$

5. Experiment

Experiments are repeated with 3 different random seeds. Regarding the uncertainty quantification methods, the sample size for training and inference is 20 and 100.

A. Dataset

In many engineering applications, data collection is very costly, both in terms of time and resource. For instance, testing a power converter component often requires accelerated aging test and destructive measurement



(a) 20 samples

(b) 100 samples

Figure 3: Metrics for uncertainty methods. Lower is better. MCDropout* = Monte Carlo Dropout with same mask. NLL = Negative Log Likelihood. RMSE = Root Mean Squared Error. Unit for Training time is $log(Minutes)$. The scale for each axis and figure is different.

procedure. Thus, the quantity of data is very limited. To simulate this context, we generate 150 data points for number of cycle, and their corresponding crack length according to Paris' model. The cycles start at 0, end at 3000, with a space of 20 in-between. We add uniform noise from the interval [-0.01, 0.01] to the generated crack lengths. The first 120 points are used for training, and the last 30 points are kept for testing. This train test split simulates the predictive maintenance use case. Regarding the physics constraint, we generate 100 samples at regular interval from the same input domain (cycle 0 to 3000).

B. Physics-informed model training

Due to the small size of the datasets, the experiments are performed with small models. Unless otherwise stated, the model architecture uses Linear layers with Tanh activation. The number of nodes are [1, 4, 4, 1]. Input and output layers have dimension = 1. There are 2 in-between hidden layers with 4 nodes each. The optimizer is Adam [23] with learning rate 0.01 and weight decay 10^{-8}. We use Ray Tune [24], a hyperparameter optimization library, to search for good values of hyperparameters. Training time is approximately 3 seconds for 10000 epochs with Intel Xeon CPU 2.30GHz.

C. Monte Carlo Dropout

The default implementation of Dropout [25] in PyTorch [26] applies a different mask for each input. Due to this behavior, the prediction interval is very sensitive to the noise in data (Figure 4). Using the same mask for all datapoints leads to a more consistent prediction interval (Figure 5).

In all repeated experiment trials, predictions from both variants of MCDropout are always too low compared to true test target. The result sometimes improves when exponentiating the prediction output, and reducing the number of training epochs to 180. However, the prediction becomes very unstable, and stray far away from the true test target very often.

Dropout probability is set at 5%. Test inference are performed with 100 samples and 20 samples (100 forward passes with dropout activated). Since the dropout probability is 1/20, taking 20 samples decreases the chance of having 0 variance. Figure 2a and Figure 2b contain calibration area plots.

D. Deep ensemble

The models in the ensemble share the same architecture as the physics-informed model. The output of each model contains 2 values: mean and variance of the prediction. Figure 6 shows the prediction interval and Figure 2d shows the miscalibration area.

E. Bayes by backprop

The prior distribution for the mean and variance of model weights is a Gaussian with mean 0 and variance 1. For the mean and variance of model bias, the prior is a uniform in the interval [-1, 1]. To enforce positive constraint on the standard deviation, we model the log standard deviation and exponentiate to recover the value. The likelihood variance σ_l^2 is set to 1. β coefficient for the KL divergence term is 10^{-6}. The number of training epochs is 10000. Our implementation of Bayes by backprop is quite naive, and leaves out many possible improvements. Figure 7 shows the prediction interval and Figure 2c shows the miscalibration area.

Table 1: Uncertainty quantification test metrics. Lower is better. Training time is in minute

Method	Negative Log Likelihood	Root Mean Squared Error	Miscalibration area	Sharpness	Training time
Ensemble 20	1.7466	0.3970	0.2912	0.3190	1.3250
Ensemble 100	1.9934	0.3995	0.3172	0.2796	6.9034
MCDropout 100	3.8454	3.2928	0.4754	1.5098	0.0080
MCDropout 20	4.4382	3.4754	0.4710	1.3365	0.0080
MCDropout* 100	3.0755	0.8375	0.4007	0.4822	0.0085
MCDropout* 20	4.5833	0.8075	0.3970	0.4706	0.0085
Bayes Backprop 20	0.7713	0.1728	0.0727	0.2679	27.5281
Bayes Backprop 100	4.1510	1.3005	0.4525	0.4069	149.0496

6. Discussion

Figure 3 and Table 1 contain a comparison of the uncertainty quantification methods. Ensemble appears to be the best overall methods, offering very competitive performance on all metrics at a reasonable training cost. Furthermore, a small ensemble size such as 20 or 10 already provides strong performance. In our experimental case, we only notice slight variations between 20 and 100 samples. Ensemble is also the most stable method, ensemble test prediction consistently matches the test target in 3 repeated trials with different random seeds.

Bayes by backprop comes in second in terms of overall performance and stability. It often provides the narrowest prediction interval. Out of 3 runs with different random seed, Bayes by backprop with 20 samples fails to match the test target once. This is the most costly method in terms of computation cost. Although our implementation leaves many rooms for improvement, the performance and stability of Bayes by backprop are insufficient to motivate the effort.

Both versions of Dropout are unstable, they fail to match the test target quite frequently. It is very intriguing to see that using the same number of training epochs as the deterministic physics-informed model causes the MCDropout models to significantly overshoot the test target. We suspect that in the small data small model regime, randomly turning off the parameters will cause the prediction to swing wildly. The same mask variant of MCDropout scores better than the different mask version.

7. Conclusion

The experiments demonstrate that we can learn from small observation dataset by incorporating physical principles. The actual implementation may require additional effort to solve multi-objective optimization problems with different variable scales. Therefore, we need appropriate rescaling calculation, and suitable balancing scheme for the coefficients of multiple loss terms with the magnitude of their gradient. Among the uncertainty quantification methods, ensemble mixture is the best overall performer.

A quantitative metric for uncertainty calibration is crucial to evaluate different models. Nevertheless, the miscalibration area metric that we use is a post-hoc analysis to assess a model after it has been trained. As an interesting direction for future work, we would like to somehow include a calibration metric into the optimization process itself. By doing so, the model is encouraged to produce well-calibrated prediction interval during the training process.

8. Acknowledgements

We would like to thank these collaborators for their help: Mohamad Nazar and Ali Ibrahim from SATIE Lab, Gustave Eiffel University, François Landes from LISN, Université Paris Saclay.

References

[1] Uwe Scheuermann and Marion Junghaenel. Limitation of power module lifetime derived from active power cycling tests. In *CIPS 2018; 10th International Conference on Integrated Power Electronics Systems*, pages 1–10. VDE, 2018.

[2] Nausicaa Dornic, Ali Ibrahim, Zoubir Khatir, Son-Ha Tran, J-P Ousten, Jeffrey Ewanchuk, and Stefan Mollov. Analysis of the degradation mechanisms occurring in the topside interconnections of igbt power devices during power cycling. *Microelectronics Reliability*, 88:462–469, 2018.

[3] P Steinhorst, Tilo Poller, and Josef Lutz. Approach of a physically based lifetime model for solder layers in power modules. *Microelectronics Reliability*, 53(9-11):1199–1202, 2013.

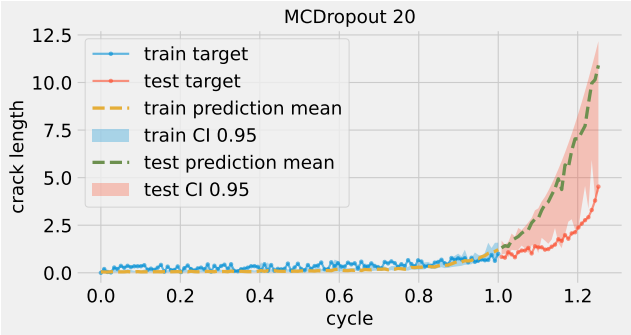
[4] Koji Sasaki, Naoko Iwasa, Toshiki Kurosu, Katsuaki Saito, Yoshihiko Koike, Yukio Kamita, and Yasushi Toyoda. Thermal and structural simulation techniques for estimating fatigue life of an igbt module. In *2008 20th International Symposium on Power Semiconductor Devices and IC's*, pages 181–184. IEEE, 2008.

[5] Louis Schuler, Ludovic Chamoin, Zoubir Khatir, Mounira Berkani, Merouane Ouhab, and Nicolas Degrenne. A reduced model based on proper generalized decomposition for the fast analysis of igbt power modules lifetime. *Journal of Electronic Packaging*, 144(3):031013, 2022.

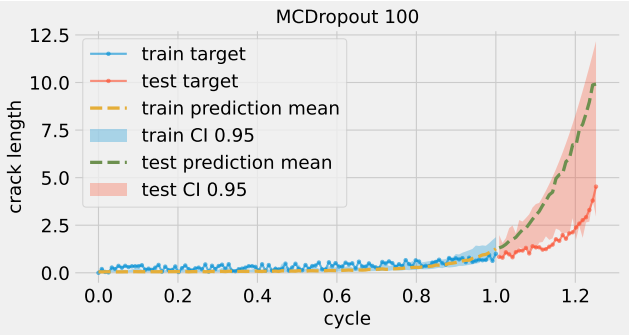
[6] Mohamad Nazar, Ali Ibrahim, Zoubir Khatir, Nicolas Degrenne, and Zeina Al Masry. Remaining useful lifetime estimation for electronic power modules using an analytical degradation model. In *Fifth European Conference of the PHM Society 2020 (The Prognostics and Health Management)*, volume 5, page 10, 2020.

[7] Nicolas Degrenne and Stefan Mollov. Diagnostics and prognostics of wire-bonded power semi-conductor modules subject to dc power cycling with physically-inspired models and particle filter. In *PHM Society European Conference*, volume 4, 2018.

[8] Yasushi Yamada, Yoshikazu Takaku, Yuji Yagi, Ikuo Nakagawa, Takashi Atsumi, Mikio Shirai, Ikuo Ohnuma, and Kiyohito Ishida. Reliability of wire-bonding and solder joint for high temperature operation of power semiconductor device. *Microelectronics Reliability*, 47(12):2147–2151, 2007.

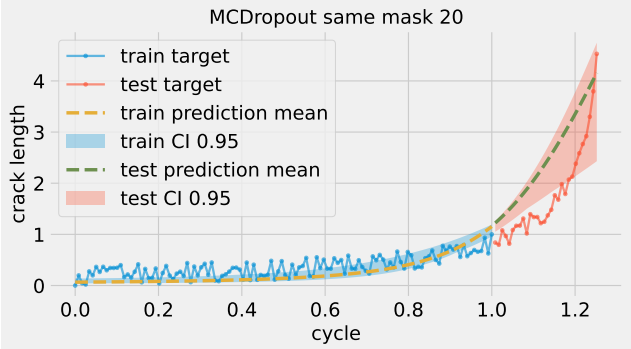


(a) 20 samples

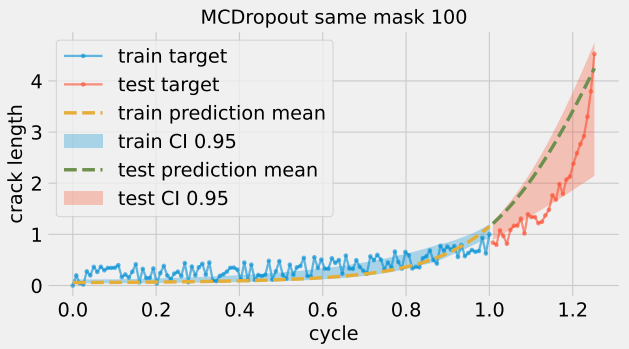


(b) 100 samples

Figure 4: Monte Carlo Dropout prediction interval with different mask for each input

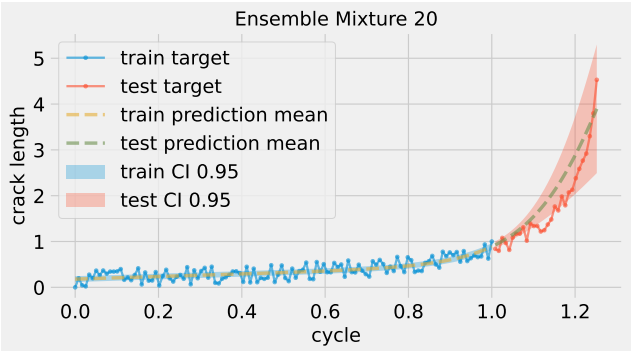


(a) 20 samples

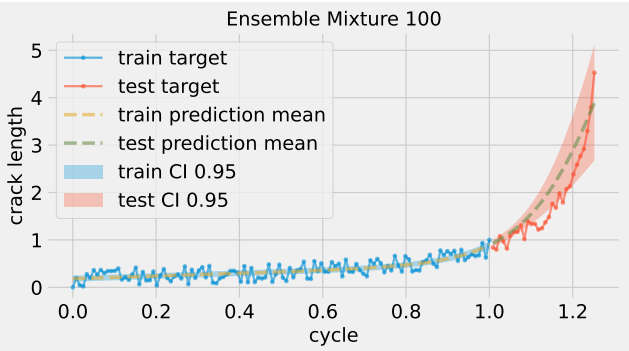


(b) 100 samples

Figure 5: Monte Carlo Dropout prediction interval with same mask for all input



(a) 20 samples



(b) 100 samples

Figure 6: Ensemble mixture prediction interval

[9] Pearl A Agyakwa, Li Yang, Elaheh Arjmand, Paul Evans, Martin R Corfield, and C Mark Johnson. Damage evolution in al wire bonds subjected to a junction temperature fluctuation of 30 k. *Journal of Electronic Materials*, 45:3659–3672, 2016.

[10] Vanessa Smet, Francois Forest, Jean-Jacques Huselstein, Frédéric Richardeau, Zoubir Khatir, Stéphane Lefebvre, and Mounira Berkani. Ageing and failure modes of igbt modules in high-temperature power cycling. *IEEE transactions on industrial electronics*, 58(10):4931–4941, 2011.

[11] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.

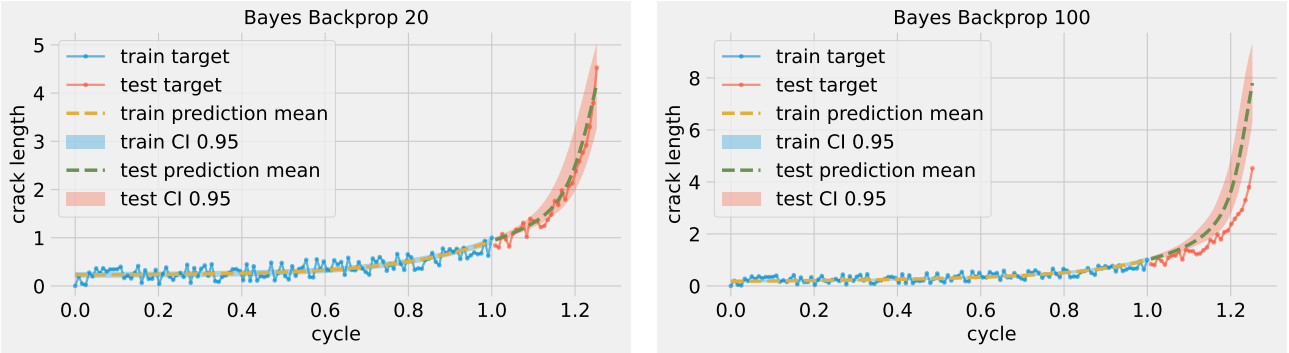
[12] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566*, 2017.

[13] Remco van der Meer, Cornelis W. Oosterlee, and Anastasia Borovykh. Optimally weighted loss functions for solving pdes with neural networks. *J. Comput. Appl. Math.*, 405(C), may 2022.

[14] Suryanarayana Maddu, Dominik Sturm, Christian L Müller, and Ivo F Sbalzarini. Inverse dirichlet weighting enables reliable training of physics informed neural networks. *Machine Learning: Science and Technology*, 3(1):015026, 2022.

[15] Paul Paris and Fazil Erdogan. A critical analysis of crack propagation laws. 1963.

[16] Arun Subramaniyan. Industrial AI: BHGE's physics-based, probabilistic deep learning using tensorflow probability



(a) 20 samples

(b) 100 samples

Figure 7: Bayes by backprop prediction inverval

- part 1, 2018. https://blog.tensorflow.org/2018/10/industrial-ai-bhges-physics-based.html.

[17] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International conference on machine learning*, pages 2796–2804. PMLR, 2018.

[18] Youngseog Chung, Ian Char, Han Guo, Jeff Schneider, and Willie Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*, 2021.

[19] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[20] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[21] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

[22] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

[25] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.