Conformal Prediction Intervals for Remaining Useful Lifetime Estimation

Alireza Javanmardi, Eyke Hüllermeier

Institute of Informatics, LMU Munich, Munich, 80799, Germany
Munich Center for Machine Learning (MCML), Munich, Germany
alireza.javanmardi@ifi.lmu.de
eyke@ifi.lmu.de

Abstract

The main objective of Prognostics and Health Management is to estimate the Remaining Useful Lifetime (RUL), namely, the time that a system or a piece of equipment is still in working order before starting to function incorrectly. In recent years, numerous machine learning algorithms have been proposed for RUL estimation, mainly focusing on providing more accurate RUL predictions. However, there are many sources of uncertainty in the problem, such as inherent randomness of systems failure, lack of knowledge regarding their future states, and inaccuracy of the underlying predictive models, making it infeasible to predict the RULs precisely. Hence, it is of utmost importance to quantify the uncertainty alongside the RUL predictions. In this work, we investigate the conformal prediction (CP) framework that represents uncertainty by predicting sets of possible values for the target variable (intervals in the case of RUL) instead of making point predictions. Under very mild technical assumptions, CP formally guarantees that the actual value (true RUL) is covered by the predicted set with a degree of certainty that can be prespecified. We study three CP algorithms to conformalize any single-point RUL predictor and turn it into a valid interval predictor. Finally, we conformalize two single-point RUL predictors, deep convolutional neural networks and gradient boosting, and illustrate their performance on the C-MAPSS datasets.

1. Introduction

Prognostics and Health Management (PHM) is devised to monitor the health state of industrial components and conduct maintenance operations when necessary. It can noticeably increase the efficiency of industrial assets by reducing their downtime, maintenance frequency, and costs accordingly. One essential element of the PHM is the Remaining Useful Lifetime (RUL) estimation, which refers to predicting the amount of time left before a system stops working as required (Jardine, Lin, & Banjevic, 2006).

In the past years, many data-driven approaches have been proposed for this problem (Y. Lei et al., 2018). From a machine learning perspective, these works attempt to solve a regression problem, that is, to discover the relationship between the condition monitoring (CM) data and the RUL. The main objective of most of these works is to predict the RUL as accurately as possible. Obviously, due to the noisiness of CM data, the stochastic behavior of systems failure, the unpredictability of systems' future states, and even the imprecision of the regression models, the RUL estimation problem is heavily affected by uncertainty (Sankararaman & Goebel, 2015). Moreover, RUL predictions will eventually affect the maintenance process, which is a delicate decision-making procedure. Hence, as a prerequisite for reliable employment in industry, it is essential to equip such predictions with a valid representation of their confidence, for instance, by answering questions of the following kind: How confident is the model with the prediction it made? What is the probability that the true RUL will actually be shorter than the predicted value, and if so, by what amount?

Uncertainty quantification (UQ) is a field in machine learning that deals with questions like these (Hüllermeier & Waegeman, 2021). In the regression problem, one way to quantify uncertainty is to predict an interval equipped with a level of confidence instead of a single value. A prediction interval provides a lower and an upper bound on the target variable, i.e., the RUL in our problem. Ideally, prediction intervals are as short as possible, and their length should represent the difficulty of the prediction (*adaptivity* property); in other words, the harder the prediction for a given data point, the higher the uncertainty and the wider the interval. More importantly, a predicted interval ought to contain the actual value of the response variable with a certain degree of probability (*coverage* property) (Angelopoulos & Bates, 2021).

https://doi.org/10.36001/IJPHM.2023.v14i2.3417

Conformal prediction (CP) is a framework for constructing prediction intervals, or, more generally, prediction regions, which has gained increasing interest in the recent past (Vovk, Gammerman, & Shafer, 2005). CP can be applied in a very versatile way and guarantees coverage property under mild technical assumptions. It is non-parametric, i.e., it makes no specific distributional assumptions on the data-generating process. Moreover, it can be put on top of any single-point RUL predictor, thereby turning it into an interval predictor. In this work, we employ the conformal prediction framework for the RUL estimation problem. More specifically, we present three CP methods and describe how to turn any single-point RUL estimator into a valid interval predictor using any of them. To the best of our knowledge, this paper is the first to investigate the CP framework for the RUL estimation problem.

The paper is organized as follows. After a brief overview of related work, the RUL estimation problem is defined in Section 3. This is followed by presenting the three conformal prediction frameworks for interval prediction. In Section 4, two single-point RUL predictors are conformalized using the CP methods, and their performance is evaluated experimentally using the C-MAPSS datasets.

2. BACKGROUND AND RELATED WORK

2.1. RUL Estimation

RUL estimation methods can be categorized into model-based and data-driven approaches (An, Kim, & Choi, 2015). Model-based approaches specify a physical degradation model according to prior domain knowledge of the system and utilize historical data to identify its parameters. Data-driven approaches employ data to discover relationships between system state and failure. Corresponding approaches range from classical machine learning methods such as support vector machines (SVM) (Benkedjouh, Medjaher, Zerhouni, & Rechak, 2013), K-nearest neighbors (KNN) (Mosallam, Medjaher, & Zerhouni, 2016), random forests (RF) (Zhang, Lim, Qin, & Tan, 2017), and gradient boosting (GB) (Zhang et al., 2017) to modern deep learning techniques, including deep belief networks (DBN) (Zhang et al., 2017), deep convolutional neural networks (DCNN) (Babu, Zhao, & Li, 2016; Li, Ding, & Sun, 2018), recurrent neural networks (RNN) (Heimes, 2008) and its variants (Y. Wu, Yuan, Dong, Lin, & Liu, 2018; Chen, Jing, Chang, & Liu, 2019; Elsheikh, Yacout, & Ouali, 2019). More recently, the problem has also been tackled by means of automated machine learning (AutoML) (Tornede, Tornede, Wever, Mohr, & Hüllermeier, 2020; Tornede, Tornede, Wever, & Hüllermeier, 2021).

2.2. Uncertainty Quantification in Regression

As mentioned by (Sankararaman & Goebel, 2015), due to the existence of multiple sources of uncertainty in prognostics, predicting a single value as an RUL is not very meaningful. Nevertheless, uncertainty quantification is relatively understudied in the field of data-driven RUL estimation. From a machine learning point of view, RUL estimation is considered a regression problem, for which four fundamental classes of interval predictors exist in the literature: Bayesian methods, ensemble methods, direct interval prediction, and CP methods (Dewolf, Baets, & Waegeman, 2022).

The most common Bayesian methods are Gaussian processes (Liu, Zhang, Liao, Wu, & Peng, 2019; Q. Wu, Ding, & Huang, 2020; Biggio, Wieland, Chao, Kastanis, & Fink, 2021) and Bayesian neural networks (Peng, Ye, & Chen, 2020; Benker, Furtner, Semm, & Zaeh, 2021). Their basic idea is to adopt a prior distribution over model parameters, which is then turned into a posterior distribution in the light of observed data. The advantage of these methods is their theoretical soundness and formal guarantees, while their main weakness is their vulnerability to model misspecification.

In ensemble methods, multiple machine learning models are trained simultaneously, and the statistics of their predictions (e.g., mean and variance) are utilized to quantify uncertainty (Rigamonti et al., 2018; Liao, Zhang, & Liu, 2018). Broadly speaking, the more the predictions diverge, the higher the uncertainty seems to be. Ensemble methods are simple and efficient but somewhat ad-hoc and difficult to interpret (e.g., they do normally not support a probabilistic interpretation).

The most well-known direct interval predictors are quantile regression models that construct intervals by providing lower and upper quantiles of response variables given their features (Zhao, Wu, Wong, Sun, & Yan, 2020). However, the intervals constructed by estimated quantiles do not guarantee coverage when dealing with a finite number of samples (Romano, Patterson, & Candès, 2019).

2.3. Conformal Prediction

Conformal prediction delivers reliable predictions in the form of sets or prediction regions (intervals in the case of regression), which are guaranteed to contain the sought target value with a predefined level of confidence. CP assumes no specific distribution for the data and works under the mild assumption of data exchangeability, which requires the joint probability distribution of a set of data points to be independent of their order. CP has originally been introduced for transductive inference in an online setting, but inductive variants have been developed later on — we refer to (J. Lei, G'Sell, Rinaldo, Tibshirani, & Wasserman, 2018) for details. Due to the computational complexity of full (transductive) CP, we focus on the inductive variant of so-called split conformal prediction in this paper.

Split CP divides the training data into subsets for proper training and calibration, using the former to fit the regression model and the latter to quantify the uncertainty of predictions (in the test set). One major drawback of the original split CP algorithm is that, in the case of regression, the prediction intervals have the same length for all data points in the test set. (Romano et al., 2019) address this issue by fitting two quantile regression models (for lower and upper bounds on the target values, respectively) and calibrating them using the calibration subset to ensure the coverage property.

All theoretical guarantees of the aforementioned approaches rely on the exchangeability assumption. However, this assumption can easily be violated, especially when dealing with ordered data such as time series. (Barber, Candes, Ramdas, & Tibshirani, 2022) tackle this issue by assigning weights to calibration data points based on their "similarity" to each data point in the test set. We will discuss this approach in more detail in Section 3.3.

3. CONFORMAL PREDICTION FOR RUL ESTIMATION

In PHM, each data instance is represented by a (possibly multivariate) time series $\mathbf{Z}_i := \{z_1^{(i)}, z_2^{(i)}, \ldots, z_{T_i}^{(i)}\}$, a collection of condition monitoring data from the moment system i starts operating up to time T_i, and a scaler F_i indicating its failure time. The RUL of instance i at time t can be computed as

$$y_t^{(i)} = F_i - t, \tag{1}$$

$t \in [T_i] := \{1, \ldots, T_i\}$.

Typically, for training data $\{(\mathbf{Z}_i, F_i)\}_{i=1}^{N_{\text{train}}}$, the time series terminate when a failure occurs, i.e., $F_i = T_i$. Such data is also referred to as *run-to-failure* data. On the other hand, for a test dataset $\{(\mathbf{Z}_k, F_k)\}_{k=1}^{N_{\text{test}}}$, the series may end at a random time before a failure occurs, i.e., $F_k \geq T_k$. Clearly, the ultimate objective is to estimate $y_{T_k}^{(k)}$ for every data point (Z_k, F_k) in the test dataset.

Depending on the regression algorithm, these training and test datasets usually need to be transformed into

$$\mathcal{D} = \left\{(x_i, y_i)\right\}_{i=1}^{N'_{\text{train}}}$$

and

$$\mathcal{D}_{\text{test}} = \left\{(x_k, y_k)\right\}_{k=1}^{N'_{\text{test}}},$$

correspondingly. In the simplest case, the transformed dataset is simply the collection of all CM data from all instances and their corresponding RULs, e.g.,

$$\mathcal{D} = \left\{(z_t^{(i)}, y_t^{(i)}) : i \in [N_{\text{train}}],\ t \in [T_i]\right\}. \tag{2}$$

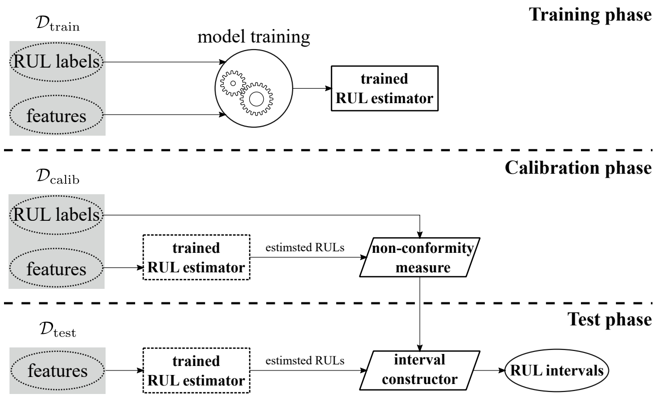


Figure 1. The general procedure of split conformal prediction for RUL estimation problem.

(Li et al., 2018) consider a time window of length L_w and stack all CM data within that interval to construct two-dimensional features. As a result, the transformed dataset can be written as

$$\mathcal{D} = \left\{ \left([z_{t-L_W+1}^{(i)}, \ldots, z_{t-1}^{(i)}, z_t^{(i)}]^\intercal, y_t^{(i)} \right) \right. \\ \left. : i \in [N_{\text{train}}],\ L_w < t \leq T_i \right\}, \tag{3}$$

where $[\cdot]^\intercal$ denotes the transpose of $[\cdot]$.

Regardless of the choice of data transformation method, in the conventional regression problem, the objective is to train a model M on $\mathcal{D}$ that mimics the relationship between instances (independent variables) x_i and targets (dependent variables) y_i, so that $\hat{y}_{\text{new}} := M(x_{\text{new}})$ is close to y_{new} for every new pair $(x_{\text{new}}, y_{\text{new}}) \in \mathcal{D}_{\text{test}}$. Alternatively, conformal prediction attempts to provide an interval $C^\alpha(x_{\text{new}}) \subset \mathbb{R}_{\geq 0}$ that contains y_{new} with a user-defined *coverage rate* $1 - \alpha$, where α is the *error rate*. Thus, the following *coverage property* holds:

$$\mathbb{P}\left(y_{\text{new}} \in C^\alpha(x_{\text{new}})\right) \geq 1 - \alpha. \tag{4}$$

In this paper, we concentrate on the split conformal prediction (SCP) framework and two of its variants, namely conformalized quantile regression (CQR) and non-exchangeable split conformal prediction (nex-SCP). The general procedure of these methods is depicted in Figure 1. It is worth noting that these methods are based on the setting of supervised learning. They can be applied to the semi-supervised setting where only a part of data is labeled (i.e., have known RUL), as long as a single-point RUL estimator can be trained using such data while putting aside a portion of labeled data for calibration. In the extreme scenario of having datasets with no known RULs, these methods are no longer applicable.

3.1. Split Conformal Prediction Framework

As its name suggests, SCP starts by randomly splitting the original training data $\mathcal{D}$ into two disjoint subsets: a proper training set $\mathcal{D}_{\text{train}}$ and a calibration set $\mathcal{D}_{\text{calib}}$. A predictive model M is then trained on the proper training set and used to obtain non-conformity scores for the examples in the calibration data. To this end, CP needs a *non-conformity measure* (aka non-conformity score function) f that takes a tuple (x, y) as input and returns a real-valued score $S = f(x, y)$ as output. The latter is meant to indicate the "strangeness" of the data point (x, y), i.e., to measure how non-conforming (x, y) is with the model M — in regression, a natural measure of non-conformity is the absolute residual error between the prediction $\hat{y}_j := M(x_j)$ and the observed outcome y_j:

$$S_j = f(x_j, y_j) = |y_j - \hat{y}_j| = |y_j - M(x_j)| \,. \tag{5}$$

Applying the non-conformity measure f to each data point in $\mathcal{D}_{\text{calib}}$, one obtains a set of non-conformity scores

$$\{S_j : (x_j, y_j) \in \mathcal{D}_{\text{calib}}\} \,.$$

Let q be the $\lceil(1 + |\mathcal{D}_{\text{calib}}|)(1 - \alpha)\rceil$ smallest value of these scores, where $\lceil\cdot\rceil$ is the ceiling function. Equivalently, q is the $(1 - \alpha)$-quantile of the empirical distribution of non-conformity scores

$$\frac{1}{|\mathcal{D}_{\text{calib}}| + 1}\,\delta_{+\infty} + \sum_{(x_j,y_j)\in\mathcal{D}_{\text{calib}}} \frac{1}{|\mathcal{D}_{\text{calib}}| + 1}\,\delta_{S_j} \,,$$

with δ_x denoting the point mass at point x. This value can be interpreted as follows: With high probability (namely, a probability of at least $1 - \alpha$), the non-conformity of a "real" data point (x, y), i.e., a data point sampled from the true underlying distribution, is $\leq q$.

The basic idea of CP, then, is to "reject" any hypothetical data point (x, y), the non-conformity of which exceeds q. In accordance with the logic of statistical hypothesis testing, the probability of erroneously rejecting a real data point (conducting a mistake of type 1) is upper-bounded by α. More specifically, given a query instance x_{new} for which a prediction is sought, a prediction region $C^{\alpha}(x_{\text{new}})$ is constructed by testing the hypothesis $(x_{new}, y_{new}) = (x_{new}, y)$ for each candidate value $y \in \mathbb{R}$, and only including those candidates for which this hypothesis cannot be rejected. In the case of regression with non-conformity measure (5), this simply leads to the interval

$$C^{\alpha}_{\text{SCP}}(x_{\text{new}}) = \left[\, \hat{y}_{\text{new}} - q, \hat{y}_{\text{new}} + q \,\right] . \tag{6}$$

The SCP procedure can be summarized as follows:

Step 1. A regression model M is trained on $\mathcal{D}_{\text{train}}$, using any regression algorithm $\mathcal{A}$:

$$M \leftarrow \mathcal{A}(\mathcal{D}_{\text{train}})$$

Step 2. The non-conformity scores S_j are computed for all $(x_j, y_j) \in \mathcal{D}_{\text{calib}}$, using the non-conformity measure (5).

Step 3. The critical non-conformity q is obtained as described above, and a prediction interval

$$C^{\alpha}_{\text{SCP}}(x_{\text{new}}) = [\hat{y}_{\text{new}} \pm q] = [M(x_{\text{new}}) \pm q]$$

is constructed for each x_{new} in the test data $\mathcal{D}_{\text{test}}$.

It can be easily observed that a lower value of α results in a higher value of q and, consequently, wider intervals. Moreover, the efficiency (i.e., narrowness) of the intervals generated by CP is dependent on the precision of the resulting point estimator M– the higher (lower) the precision of the estimator, the more (less) efficient the intervals become.

One limitation of this procedure is a lack of adaptivity: The length of prediction intervals is always the same, namely $2q$, regardless of the query instance x_{new}. In contrast, one would intuitively expect that the width of an interval is adapted to the "difficulty" of the prediction. One way to tackle this problem is to modify the non-conformity measure to account for the difficulty of the data points (J. Lei et al., 2018). For example, a *normalized* non-conformity measure can be defined as

$$S_j = \frac{|y_j - \hat{y}_j|}{\sigma(x_j)} \tag{7}$$

for calibration data $(x_j, y_j) \in \mathcal{D}_{\text{calib}}$, where σ is another regression model trained on $\{(x_i, |y_i - \hat{y}_i|) : (x_i, y_i) \in \mathcal{D}_{\text{train}}\}$. Given q as the $\lceil(1 + |\mathcal{D}_{\text{calib}}|)(1 - \alpha)\rceil$ smallest value of the normalized non-conformity scores, the prediction interval at x_{new} is formed as

$$\left[\, \hat{y}_{\text{new}} \pm q\sigma(x_{\text{new}}) \,\right] . \tag{8}$$

In the following section, we introduce another approach that achieves adaptivity in a different way.

3.2. Conformalized Quantile Regression Framework

The general procedure of the CQR approach is similar to SCP except for the choice of the regression algorithm and the non-conformity measure. Conventional regression algorithms estimate the conditional mean of the response variable given its features (i.e., $\mathbb{E}[Y|X = x]$). In $\tau-$quantile regression, on the other hand, we are interested in estimating a conditional quantile of the response variable given its features (i.e., $Q_\tau(Y|X = x)$[1]). This can be accomplished by training a regression model on a specific loss function, the *pinball loss*,

[1]$Q_\tau(Y|X = x) := \inf\{y \in \mathbb{R} : F(y|X = x) \geq \tau\}$ where $F(y|X = x) := \mathbb{P}(Y \leq y|X = x)$ is the conditional distribution of Y given $X = x$.

which is defined as follows:

$$PL_\tau(y,\hat{y}) := \begin{cases} \tau(y-\hat{y}) & \text{if } y > \hat{y} \\ (1-\tau)(\hat{y}-y) & \text{otherwise} \end{cases} . \tag{9}$$

The procedure of CQR is as follows:

Step 1. Two quantile regression models are fitted on $\mathcal{D}_{\text{train}}$ using any quantile regression algorithm $\mathcal{A}$, one with $\tau_{\text{low}} = \alpha$ and the other one with $\tau_{\text{high}} = 1-\alpha$:

$$\hat{Q}_{\tau_{\text{low}}}, \hat{Q}_{\tau_{\text{high}}} \leftarrow \mathcal{A}(\mathcal{D}_{\text{train}})$$

Step 2. For measuring the non-conformity, the following scoring function is used:

$$S_j = \max\left\{\hat{Q}_{\tau_{\text{low}}}(x_j) - y_j, y_j - \hat{Q}_{\tau_{\text{high}}}(x_j)\right\}. \tag{10}$$

Step 3. Let q be the $\lceil(1+|\mathcal{D}_{\text{calib}}|)(1-\alpha)\rceil$ smallest value of the non-conformity scores on the calibration data. The prediction interval for a new test instance x_{new} is then constructed as follows:

$$C^\alpha_{\text{CQR}}(x_{\text{new}}) = \left[\,\hat{Q}_{\tau_{\text{low}}}(x_{\text{new}}) - q, \hat{Q}_{\tau_{\text{high}}}(x_{\text{new}}) + q\,\right]. \tag{11}$$

The scoring function (10) has meaningful properties: The score is positive if the actual response variable y_j is outside the interval $[\hat{Q}_{\tau_{\text{low}}}(x_j), \hat{Q}_{\tau_{\text{high}}}(x_j)]$, accounting undercoverage; otherwise, if y_j is covered by $[\hat{Q}_{\tau_{\text{low}}}(x_j), \hat{Q}_{\tau_{\text{high}}}(x_j)]$, the score is non-positive, thereby handling the overcoverage problem (Romano et al., 2019). Furthermore, the length of $C^\alpha_{\text{CQR}}(x_{\text{new}})$ varies with x_{new}, which represents adaptivity. It is proved in (Romano et al., 2019, Theorem 1) that under the exchangeability assumption of the data points in $\mathcal{D}_{\text{calib}} \cup \mathcal{D}_{\text{test}}$, the intervals given in (11) satisfy the coverage property (4).

3.3. Non-exchangeable Split Conformal Prediction

The coverage property of CP, namely,

$$\mathbb{P}\Big(y_{\text{new}} \in C^\alpha(x_{\text{new}})\Big) \geq 1-\alpha,$$

is guaranteed under relatively mild technical assumptions (J. Lei et al., 2018). One important assumption that needs to be satisfied, however, is the exchangeability of the underlying data-generating process.

Assumption 1 (Exchangeability)**.** *Random variables $V_1, V_2, \ldots, V_m$ are called exchangeable if their joint distribution does not depend on their order:*

$$\mathbb{P}(V_1, V_2, \ldots, V_m) = \mathbb{P}(V_{\pi(1)}, V_{\pi(2)}, \ldots, V_{\pi(m)})$$

for any permutation $\pi : [m] \to [m]$.

In the case of conformal prediction, the random variables of interest include the data $v_j = (x_j, y_j)$ in the calibration data $\mathcal{D}_{\text{calib}}$ and the new test case $v_{new} = (x_{new}, y_{new})$.

Exchangeability is weaker than the common assumption of independent and identically distributed (i.i.d.) data, i.e., the former implies the latter but not the other way around. Nevertheless, in the case of data with a temporal component, even exchangeability will probably be violated. In the following, we describe a method proposed by (Barber et al., 2022), which modifies the SCP framework to give valid intervals even when the exchangeability assumption does not hold.

Consider a new data point $(x_{\text{new}}, y_{\text{new}})$ in the test set and assume that data points in $\mathcal{D}_{\text{calib}} \cup \{(x_{\text{new}}, y_{\text{new}})\}$ are not exchangeable. Suppose we have an idea about the underlying similarity between the distributions of $(x_{\text{new}}, y_{\text{new}})$ and the points in $\mathcal{D}_{\text{calib}}$. In that case, we can assign weights $w_j \in [0,1]$ for every (x_j, y_j) in $\mathcal{D}_{\text{calib}}$, with higher weights indicating higher similarity. For instance, consider a time series $(x_1,y_1),(x_2,y_2),\ldots,(x_t,y_t),(x_{t+1},y_{t+1})$, with the first t points being the calibration data and (x_{t+1}, y_{t+1}) the test point. It would then be natural to choose weights $w_1 \leq w_2 \leq \ldots \leq w_t$ such that the more recent data points have greater weights than the less recent ones.

Given calibration data $(x_j, y_j) \in \mathcal{D}_{\text{calib}}$ with non-conformity scores S_j according to (5) and weights w_j, we define normalized weights

$$\tilde{w}_j = \frac{w_j}{1 + \sum_{j \in \mathcal{D}_{\text{calib}}} w_j}. \tag{12}$$

These normalized weights can be used to modify the empirical distribution of non-conformity scores as follows:

$$\sum_{(x_j,y_j)\in\mathcal{D}_{\text{calib}}} \tilde{w}_j \delta_{S_j} + \tilde{w}_{+\infty}\delta_{+\infty}, \tag{13}$$

with δ_x being the point mass at x, and

$$\tilde{w}_{+\infty} = \frac{1}{1 + \sum_{j \in \mathcal{D}_{\text{calib}}} w_j}.$$

The procedure of nex-SCP is similar to SCP except for the last step. This time, $q(x_{\text{new}})$ needs to be calculated for every $(x_{\text{new}}, y_{\text{new}}) \in \mathcal{D}_{\text{test}}$ separately and is defined as the $(1-\alpha)$-quantile of the empirical distribution of non-conformity scores given in (13). Accordingly, the prediction interval at x_{new} is constructed as follows:

$$C^\alpha_{\text{nex-SCP}}(x_{\text{new}}) = \left[\,\hat{y}_{\text{new}} - q(x_{\text{new}}), \hat{y}_{\text{new}} + q(x_{\text{new}})\,\right]. \tag{14}$$

Without the assumption of exchangeability, the level of coverage that can be guaranteed is reduced compared to the exchangeable case; we refer to (Barber et al., 2022, Theorem 2a) for more details on the theoretical properties of this method.

4. EXPERIMENTS

The main focus of this paper is on the CP algorithms and how to conformalize any single-point RUL estimator, turning it into a reliable interval predictor. For this purpose, we employ two existing single-point RUL estimators, Deep Convolutional Neural Networks (**DCNN**) and Gradient Boosting (**GB**), and conformalize them using **SCP**, SCP with normalized non-conformity measure (**SCP+NNM**), **nex-SCP**, nex-SCP with normalized non-conformity measure (**nex-SCP+NNM**), and **CQR**. Our implementation code is publicly available on GitHub[2] to enable the reproducibility of the presented results.

4.1. C-MAPSS Data

C-MAPSS is a software written in the MATLAB-Simulink environment, which is capable of simulating a large commercial turbofan engine with various tunable input parameters to specify numerous operational profiles, environmental conditions, initial wear degrees, degradations, etc. (Saxena, Goebel, Simon, & Eklund, 2008) ran this simulation environment multiple times with different parameter values while collecting noisy data from many units, including twenty-one sensor measurements and three operational settings. They provided four datasets of multivariate time series, where each dataset consists of a training and a test set. At the beginning of each time series, the engine operates normally, and at a random point during the series, it starts to degrade. For training sets, a time series terminates when the engine failure occurs, and the RUL at each time step is defined as the number of time steps left until the end of the series. For the test sets, a series ends at a random time before the failure, while the actual RUL for its last time step is provided.

4.1.1. Data Preprocessing

The details of the four datasets of C-MAPSS are provided in Table 1. For datasets #1 and #3, where the operating conditions do not vary with time (stationary cases), we employ a min-max scaler to map the sensor measurements into the range of $[-1, 1]$. On the other hand, for datasets #2 and #4, the operating conditions alter between 6 different operating modes (nonstationary cases). However, the information on engines' operating conditions is only available through three continuous-valued columns in the data – operational settings 1 to 3. To transform these continuous values into discrete operating modes, we used the K-means clustering algorithm with $K = 6$. Furthermore, to reduce the effect of nonstationary operating conditions from the sensor measurements, we used six separate min-max scalers (with the range $[-1, 1]$) to normalize sensor measurements within each operating mode. This technique can only be utilized when the learner has access to the operating conditions and the number of operating modes is finite and known beforehand. If any of the conditions are violated, this approach is no longer applicable in its present form.

Table 1. Description of the four C-MAPSS datasets.

Dataset	**# 1**	**# 2**	**# 3**	**# 4**
# Training instances	100	260	100	249
# Test instances	100	259	100	248
# Operating conditions	1	6	1	6
# Fault modes	1	1	2	2

Moreover, seven of the twenty-one sensor measurements have zero (or close to zero) variances and provide no useful information. Hence, we remove these seven sensors with the indices 1, 5, 6, 10, 16, 18, and 19.

A piecewise linear definition of RUL labels (aka rectified labels) exists for C-MAPSS datasets that limits the maximum value of the RUL (Heimes, 2008). These rectified labels can be easily defined by modifying (1) as follows:

$$y_t^{(i)} = \max\left(\mathrm{RUL}_{\max}, F_i - t\right) \tag{15}$$

for all i and $t \in [T_i]$, where $\mathrm{RUL}_{\max}$ is a fixed value chosen on the basis of the observations. The intuition behind this definition is that the system's degradation begins after a certain degree of usage. Similar to (Li et al., 2018), we also set $\mathrm{RUL}_{\max}$ to 125 for all four datasets.

4.2. Learning Algorithms

The following single-point RUL estimators are used in our experiments:

- Deep Convolutional Neural Networks (**DCNN**) proposed by (Li et al., 2018): For this method, the original datasets are transferred using the windowing technique to be in the form of (3). The window lengths are set to be 30, 20, 30, and 15 for the datasets #1 to #4, respectively. During test time, we only use one data point corresponding to the last recorded cycle for each engine unit.
  We use the same architecture and parameters as in (Li et al., 2018). The network is constructed by stacking four identical convolution layers, each with ten filters of size 10×1, followed by another convolution layer with a single filter of size 3×1, a Flatten layer, a fully connected layer with 100 neurons, and a single neuron that outputs the RUL estimation. Zero-padding is used in convolutional layers to keep the data dimension unchanged. Except for the last single neuron that uses a linear activation function, the others use *tanh*. In order to prevent overfitting, the Dropout technique with the rate of 0.5 is used right after the Flatten layer. The Adam optimizer is used for minimizing the mean squared error (MSE) as a loss

[2] https://github.com/alireza-javanmardi/conformal-RUL-intervals

function. Different from (Li et al., 2018), for the CQR framework, we replace the MSE loss with the pinball loss function to obtain quantiles, as explained in Section 3.2. Models are trained for 250 epochs with a batch size of 512 samples. The learning rate is set to 0.001 for the first 200 epochs and 0.0001 for the last 50.

- Gradient Boosting (**GB**): The transformed datasets are in the form of (2), e.g., the collection of all condition monitoring data from all instances and their corresponding rectified RULs. Once again, only one data point corresponding to the last recorded cycle for each engine unit is used at test time.

  The default setting of the `HistGradientBoostingRegressor` from scikit-learn (Pedregosa et al., 2011) is adopted for training the model(s). Like in the case of DCNN, the MSE loss function is replaced by the pinball loss for quantile regression in the CQR framework.

4.3. Results and Discussion

For each of the C-MAPSS datasets and each learning algorithm, we first divide the training data into (proper) training and calibration sets such that the data points from the same engine unit end in the same subset. The proportion of the data to be included in the calibration set is fixed at 10%. Using the learning algorithm, we train a regression model and eight quantile regression models for the quantile set $\{0.10, 0.15, 0.20, 0.25, 0.75, 0.80, 0.85, 0.90\}$. This way, we are able to perform conformal prediction using multiple miscoverage rates $\{0.10, 0.15, 0.20, 0.25\}$. For CP frameworks with normalized non-conformity measure (i.e., SCP+NNM and nex-SCP+NNM), we realize σ by training a Random Forest regressor using the default setting of the scikit-learn for `RandomForestRegressor`. For non-exchangeable methods (i.e., nex-SCP and nex-SCP+NNM), we define the set of weights for every data point $(x_{\text{new}}, y_{\text{new}})$ in the test set as

$$w_j = 0.99^{|t(y_{\text{new}})-t(y_j)|},$$

$\forall (x_j, y_j) \in \mathcal{D}_{\text{calib}}$, where $t(y)$ denotes the time index of recording y (Barber et al., 2022).

Figure 2 illustrates the prediction intervals with $\alpha = 0.1$ for the test units of C-MAPSS dataset #1. By looking at the dashed lines of this figure, one can realize that single-point RUL predictions get more accurate for data points closer to failure times[3]. Hence, these data points can be considered easier than those with actual RULs far from zero, and we can expect to observe shorter prediction intervals for them, which is exactly how adaptive methods should behave. Moreover, since DCNN works better than GB in terms of single-point prediction preciseness, its prediction intervals are also shorter on average.

We run the experiments with 15 random train-calibration splits for each C-MAPSS dataset and each learning algorithm. In Figure 3, the average coverages and interval widths are shown for all four datasets using DCNN as the underlying learning model. Similar results are provided in Figure 4 when GB is the learning model. The horizontal dashed lines indicate the nominal coverage levels (i.e., 0.75, 0.80, 0.85, and 0.90). On average, non-exchangeable frameworks outperform other frameworks in terms of average coverages, while CQR has the best performance in terms of average interval widths. As expected, the average interval widths decrease as the miscoverage rate increases. Using the normalized nonconformity measure often improves the results, especially for nonstationary cases.

It should be noted that the coverage guarantee of conformal prediction is marginal and only holds on average across all data points. This implies that the coverage might exceed the expected value in certain regions of RUL while falling below the expected value in other regions. Therefore, we cannot draw any conclusions about the validity of the prediction intervals at different regions of RUL.

5. CONCLUSION

This paper makes the conformal prediction framework amenable to the remaining useful lifetime estimation problem. This allows for specifying the uncertainty of RUL predictions by constructing prediction intervals that include the actual value with a user-defined probability. We reviewed some of the existing CP frameworks and showed how to turn any single-point RUL estimator into an interval predictor. Using deep convolutional neural networks and gradient-boosting algorithms as underlying regression models, we confirmed the validity and evaluated the effectiveness of CP frameworks on the popular C-MAPSS datasets.

Needless to say, there is still scope for improvement. First, the performance of conformal prediction heavily depends on the precision of the underlying single-point predictor. Indeed, a more precise model results in lower errors on calibration data and, accordingly, a smaller quantile q. This results in shorter prediction intervals while satisfying the coverage property under the exchangeability assumption. Therefore, it is always beneficial to increase the accuracy of regression models for the RUL estimation problem. Moreover, there is also still room for developing better conformal prediction methods for the non-exchangeable case or even a framework specifically tailored to the RUL estimation problem.

[3]The $0.5-$quantile regression model (aka median regression) is used as the single-point RUL estimator in the CQR framework.

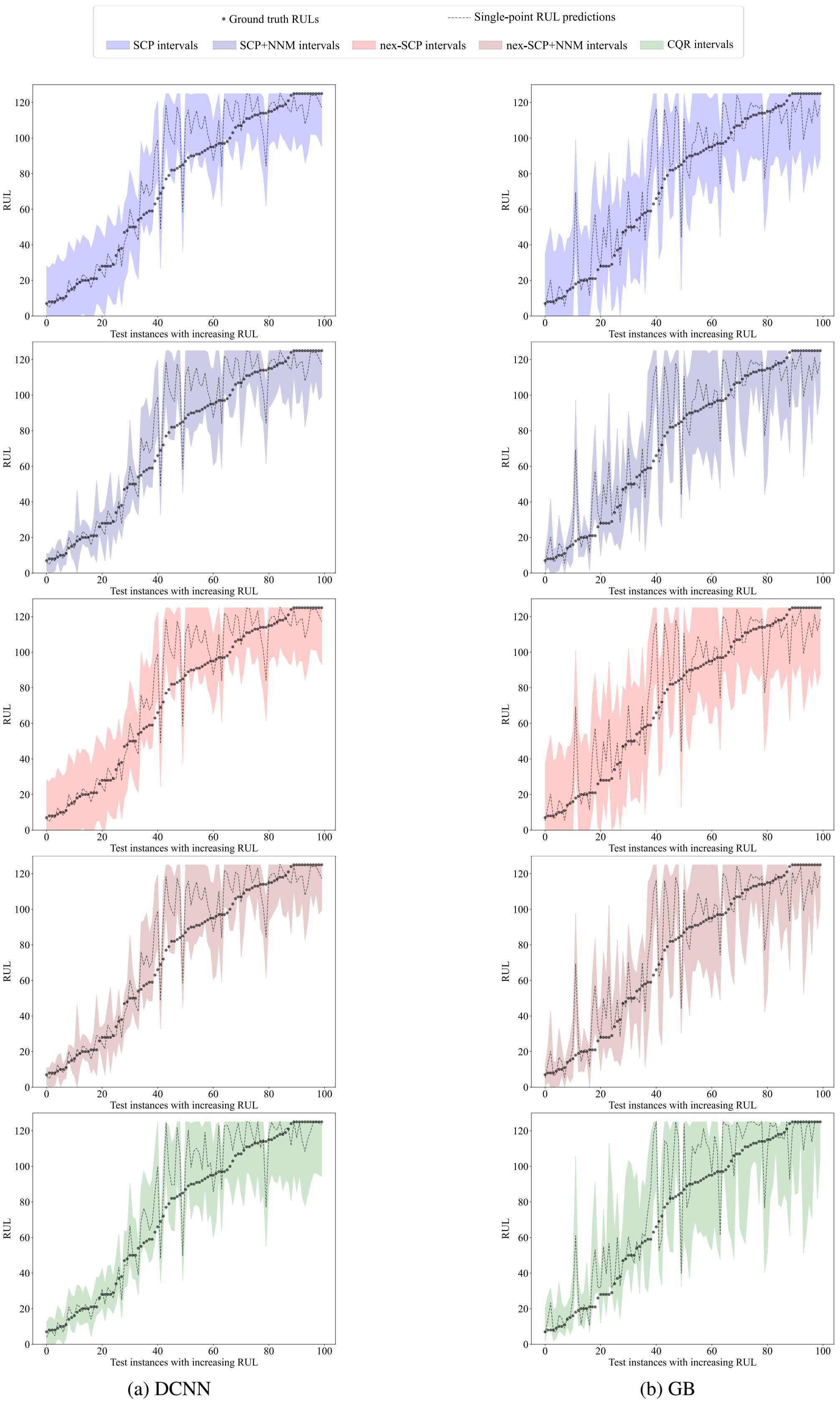
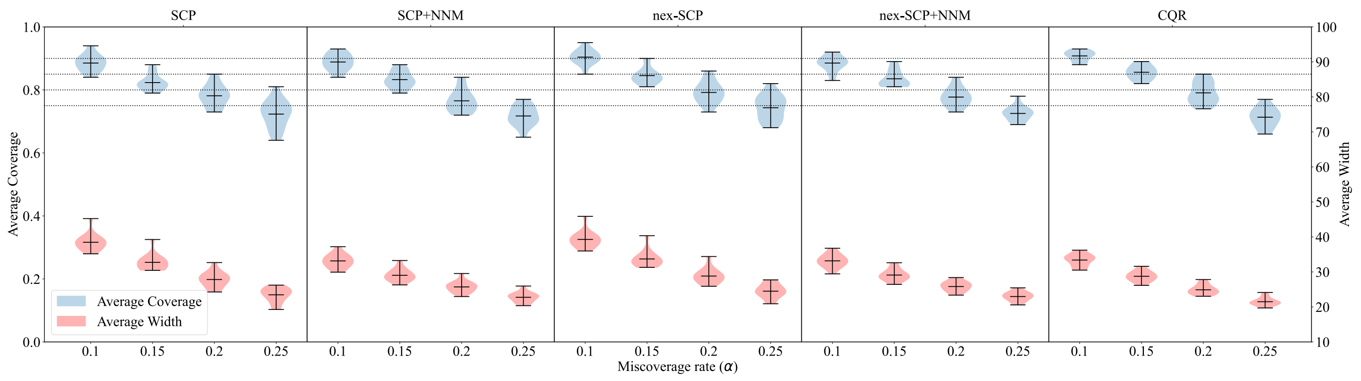
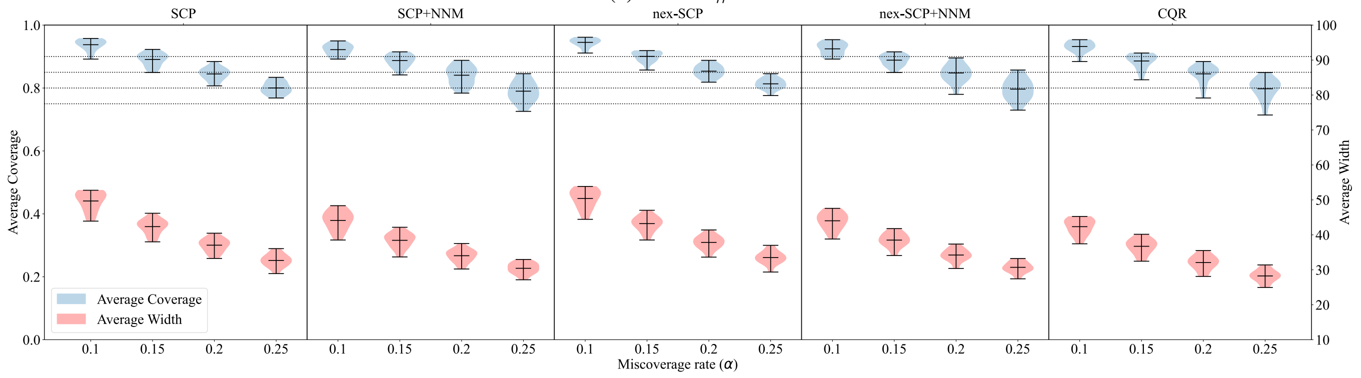


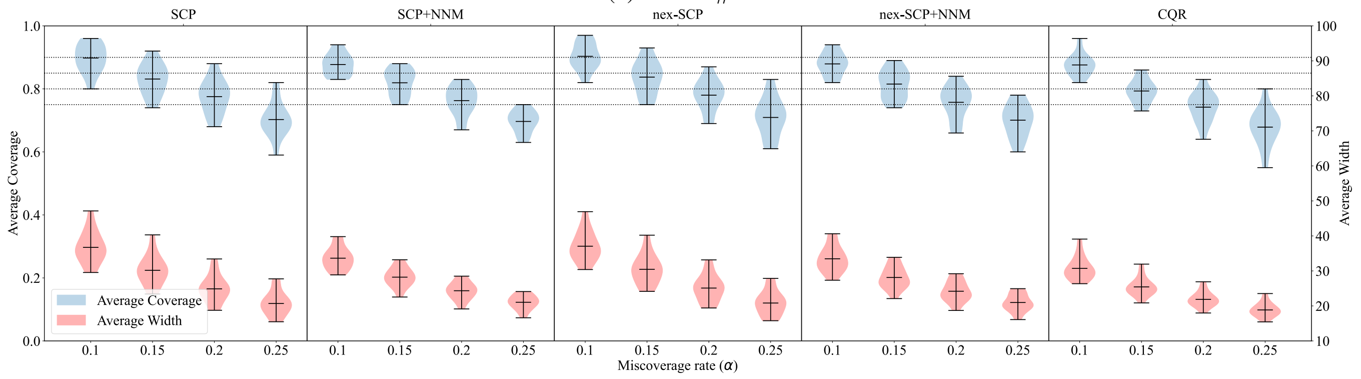
Figure 2. Sorted RUL labels of test instances of dataset #1 with their predicted intervals from five CP methods.



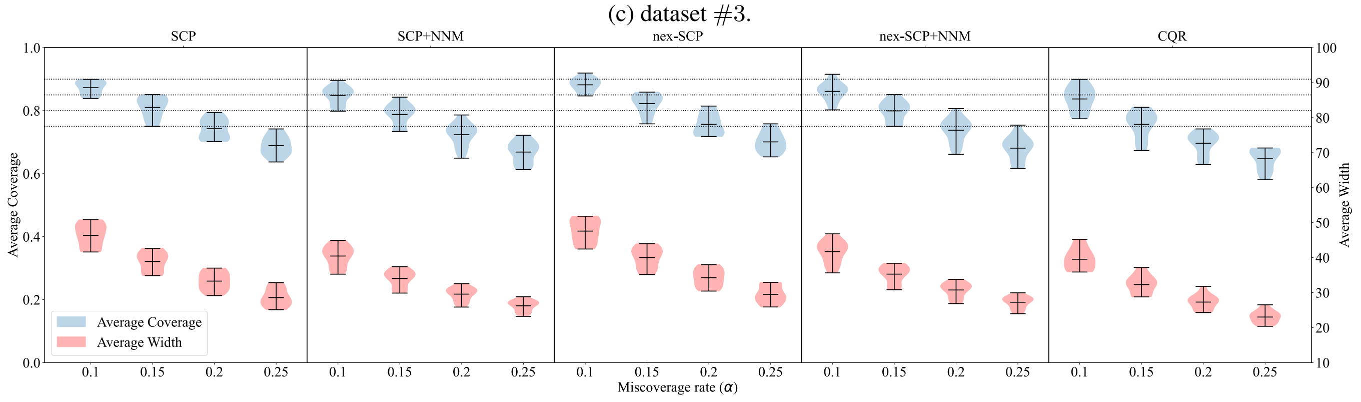
(a) dataset #1.



(b) dataset #2.

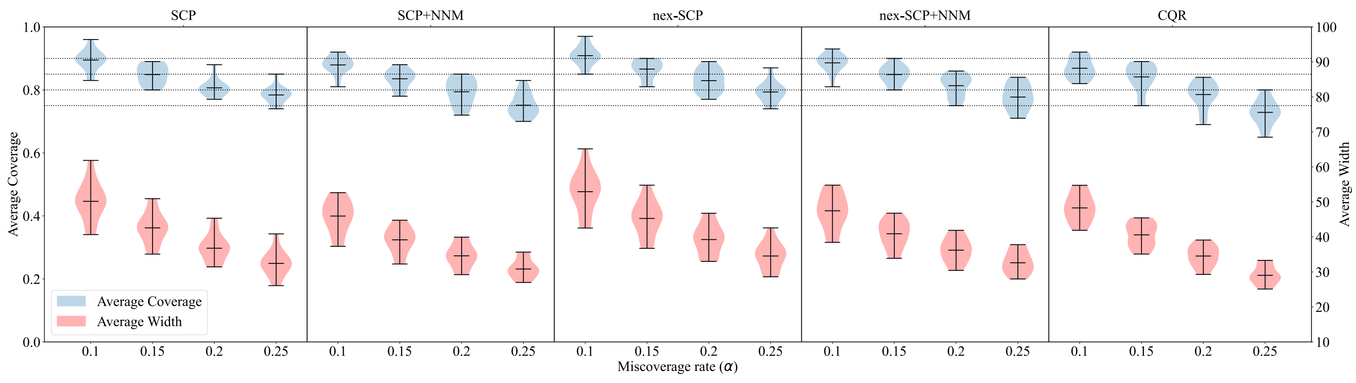


(c) dataset #3.

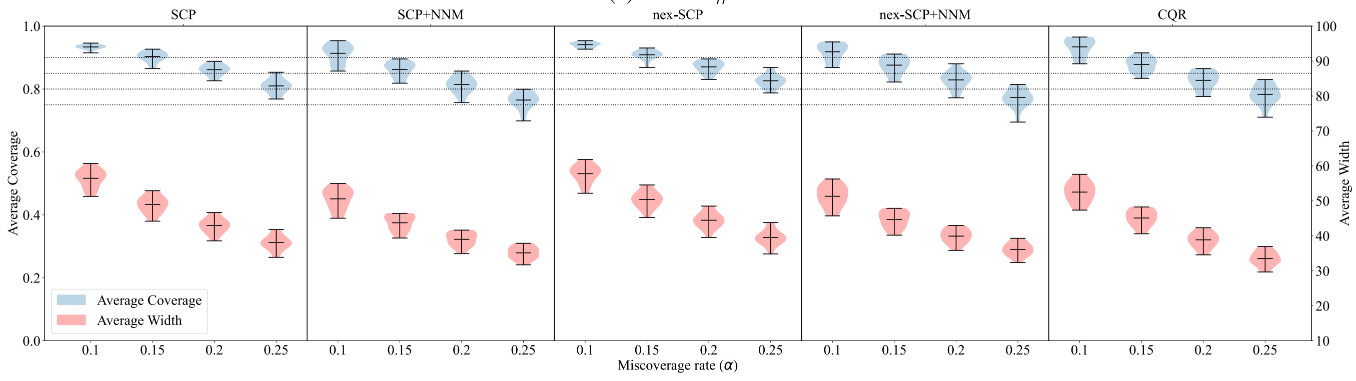


(d) dataset #4.

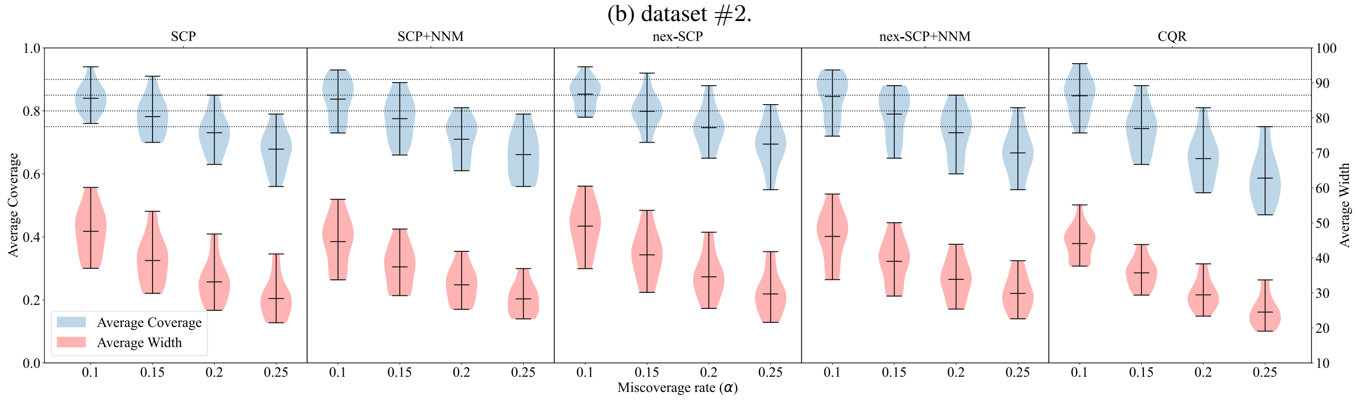
Figure 3. Average coverage and average prediction interval width of different CP frameworks for C-MAPSS datasets, using DCNN as the underlying regression model.



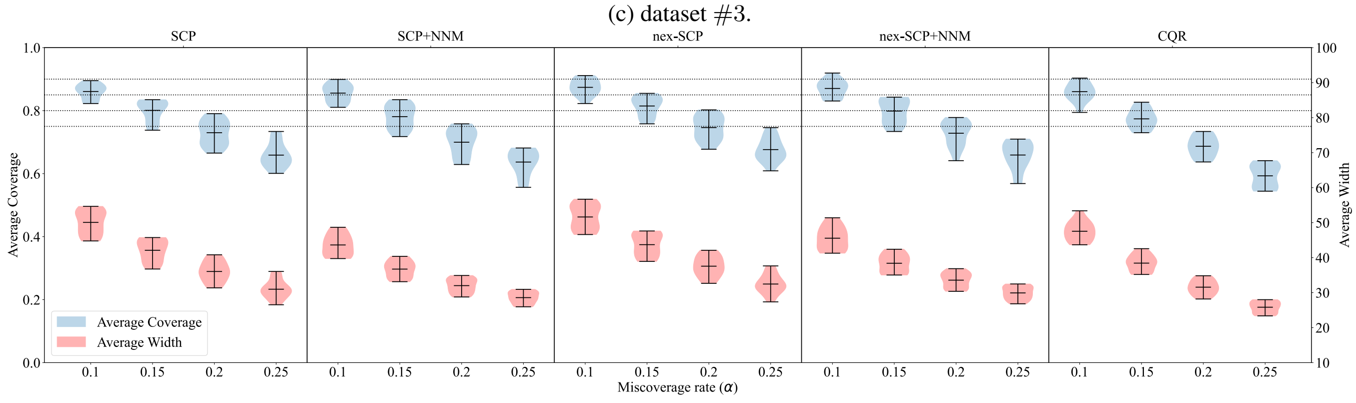
(a) dataset #1.



(b) dataset #2.



(c) dataset #3.



(d) dataset #4.

Figure 4. Average coverage and average prediction interval width of different CP frameworks for C-MAPSS datasets, using GB as the underlying regression model.

ACKNOWLEDGMENT

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): Project number 451737409.

REFERENCES

An, D., Kim, N. H., & Choi, J. H. (2015). *Practical options for selecting data-driven or physics-based prognostics algorithms with reviews* (Vol. 133). Retrieved from http://dx.doi.org/10.1016/j.ress.2014.09.014 doi: 10.1016/j.ress.2014.09.014

Angelopoulos, A. N., & Bates, S. (2021). A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. *arXiv*. Retrieved from http://arxiv.org/abs/2107.07511

Babu, G. S., Zhao, P., & Li, X. L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (Vol. 9642, pp. 214–228). Springer Verlag. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-32025-0_14 doi: 10.1007/978-3-319-32025-0_14

Barber, R. F., Candes, E. J., Ramdas, A., & Tibshirani, R. J. (2022). Conformal prediction beyond exchangeability. *arXiv*. Retrieved from http://arxiv.org/abs/2202.13415

Benkedjouh, T., Medjaher, K., Zerhouni, N., & Rechak, S. (2013). Remaining useful life estimation based on nonlinear feature reduction and support vector regression. *Engineering Applications of Artificial Intelligence*, *26*(7), 1751–1760. Retrieved from http://dx.doi.org/10.1016/j.engappai.2013.02.006 doi: 10.1016/j.engappai.2013.02.006

Benker, M., Furtner, L., Semm, T., & Zaeh, M. F. (2021, oct). Utilizing uncertainty information in remaining useful life estimation via Bayesian neural networks and Hamiltonian Monte Carlo. *Journal of Manufacturing Systems*, *61*, 799–807. doi: 10.1016/j.jmsy.2020.11.005

Biggio, L., Wieland, A., Chao, M. A., Kastanis, I., & Fink, O. (2021). Uncertainty-aware Remaining Useful Life predictor. *arXiv*. Retrieved from http://arxiv.org/abs/2104.03613

Chen, J., Jing, H., Chang, Y., & Liu, Q. (2019). Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process. *Reliability Engineering and System Safety*, *185*(January), 372–382. doi: 10.1016/j.ress.2019.01.006

Dewolf, N., Baets, B. D., & Waegeman, W. (2022). Valid prediction intervals for regression problems. *Artificial Intelligence Review*. Retrieved from https://doi.org/10.1007/s10462-022-10178-5 doi: 10.1007/s10462-022-10178-5

Elsheikh, A., Yacout, S., & Ouali, M. S. (2019). Bidirectional handshaking LSTM for remaining useful life prediction. *Neurocomputing*, *323*, 148–156. Retrieved from https://doi.org/10.1016/j.neucom.2018.09.076 doi: 10.1016/j.neucom.2018.09.076

Heimes, F. O. (2008). Recurrent neural networks for remaining useful life estimation. *2008 International Conference on Prognostics and Health Management, PHM 2008*. doi: 10.1109/PHM.2008.4711422

Hüllermeier, E., & Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, *110*(3), 457–506. Retrieved from https://doi.org/10.1007/s10994-021-05946-3 doi: 10.1007/s10994-021-05946-3

Jardine, A. K., Lin, D., & Banjevic, D. (2006, oct). *A review on machinery diagnostics and prognostics implementing condition-based maintenance* (Vol. 20) (No. 7). doi: 10.1016/j.ymssp.2005.09.012

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., & Wasserman, L. (2018). Distribution-Free Predictive Inference for Regression. *Journal of the American Statistical Association*, *113*(523), 1094–1111. Retrieved from https://doi.org/10.1080/01621459.2017.1307116 doi: 10.1080/01621459.2017.1307116

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018, may). *Machinery health prognostics: A systematic review from data acquisition to RUL prediction* (Vol. 104). Academic Press. doi: 10.1016/j.ymssp.2017.11.016

Li, X., Ding, Q., & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering and System Safety*, *172*, 1–11. Retrieved from http://www.elsevier.com/open-access/userlicense/1.0/ doi: 10.1016/j.ress.2017.11.021

Liao, Y., Zhang, L., & Liu, C. (2018, aug). Uncertainty Prediction of Remaining Useful Life Using Long Short-Term Memory Network Based on Bootstrap Method. In *2018 ieee international conference on prognostics and health management, icphm 2018.* Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/ICPHM.2018.8448804

Liu, C., Zhang, L., Liao, Y., Wu, C., & Peng, G. (2019). Multiple Sensors Based Prognostics with Prediction Interval Optimization via Echo State Gaussian Process. *IEEE Access*, *7*, 112397–112409. doi: 10.1109/ACCESS.2019.2925634

Mosallam, A., Medjaher, K., & Zerhouni, N. (2016). Data-driven prognostic method based on Bayesian approaches for direct remaining useful life prediction. *Journal of Intelligent Manufacturing*, *27*(5), 1037–1048. doi: 10.1007/s10845-014-0933-4

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. Retrieved from http://scikit-learn.sourceforge.net.

Peng, W., Ye, Z. S., & Chen, N. (2020). Bayesian Deep-Learning-Based Health Prognostics Toward Prognostics Uncertainty. *IEEE Transactions on Industrial Electronics*, *67*(3), 2283–2293. doi: 10.1109/TIE.2019.2907440

Rigamonti, M., Baraldi, P., Zio, E., Roychoudhury, I., Goebel, K., & Poll, S. (2018). Ensemble of optimized echo state networks for remaining useful life prediction. *Neurocomputing*, *281*, 121–138. Retrieved from https://doi.org/10.1016/j.neucom.2017.11.062 doi: 10.1016/j.neucom.2017.11.062

Romano, Y., Patterson, E., & Candès, E. J. (2019). Conformalized quantile regression. In *Advances in neural information processing systems* (Vol. 32). Retrieved from https://github.com/yromano/cqr.

Sankararaman, S., & Goebel, K. (2015). Uncertainty in prognostics and systems health management. *International Journal of Prognostics and Health Management*, *6*, 1–14. doi: 10.36001/ijphm.2015.v6i4.2319

Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management, phm 2008.* doi: 10.1109/PHM.2008.4711414

Tornede, T., Tornede, A., Wever, M., & Hüllermeier, E. (2021). Coevolution of remaining useful lifetime estimation pipelines for automated predictive maintenance. In *Gecco 2021 - proceedings of the 2021 genetic and evolutionary computation conference* (pp. 368–376). Retrieved from https://doi.org/10.1145/3449639.3459395 doi: 10.1145/3449639.3459395

Tornede, T., Tornede, A., Wever, M., Mohr, F., & Hüllermeier, E. (2020). AutoML for Predictive Maintenance: One Tool to RUL Them All. In *Communications in computer and information science* (Vol. 1325, pp. 106–118). Retrieved from https://doi.org/10.1007/978-3-030-66770-2_8 doi: 10.1007/978-3-030-66770-2_8

Vovk, V., Gammerman, A., & Shafer, G. (2005). *Algorithmic learning in a random world*. Springer-Verlag. doi: 10.1007/b106715

Wu, Q., Ding, K., & Huang, B. (2020, oct). Approach for fault prognosis using recurrent neural network. *Journal of Intelligent Manufacturing*, *31*(7), 1621–1633. Retrieved from https://link.springer.com/article/10.1007/s10845-018-1428-5 doi: 10.1007/s10845-018-1428-5

Wu, Y., Yuan, M., Dong, S., Lin, L., & Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*, *275*, 167–179. Retrieved from http://dx.doi.org/10.1016/j.neucom.2017.05.063 doi: 10.1016/j.neucom.2017.05.063

Zhang, C., Lim, P., Qin, A. K., & Tan, K. C. (2017, oct). Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2306–2318. doi: 10.1109/TNNLS.2016.2582798

Zhao, Z., Wu, J., Wong, D., Sun, C., & Yan, R. (2020). Probabilistic Remaining Useful Life Prediction Based on Deep Convolutional Neural Network. *SSRN Electronic Journal*. Retrieved from https://github.com/ZhaoZhibin/Probabilistic_RUL_Prediction. doi: 10.2139/ssrn.3717738