Methods to Improve the Prognostics of Time-to-Failure Models

Edward Baumann[1], Pedro A. Forero[2], Gray Selby[3], and Charles Hsu[4]

[1,3,4] *Trident Systems Inc., Fairfax, VA, 22030, USA*
{*edward.baumann; gray.selby; hsu*}*@tridsys.com*

[2] *Naval Information Warfare Center Pacific, San Diego, CA, 92152, USA*
pedro.a.forero@navy.mil

ABSTRACT

Autonomous and autonomic systems have started to develop machine learning (ML) methods for prognostics and health management (PHM) directly at the platform level. Remaining-useful-life (RUL) estimation, also known as Time-to-failure (TTF) estimation, using streaming sensor data is critical for PHM as it can help to decide and schedule appropriate courses of action (COAs). This work casts the RUL-estimation problem as a classification problem over a finite-time horizon. Rather than using a winner-take-all method to develop a RUL estimator, we propose a top-K estimator that considers the RUL values corresponding to the K-largest probabilities yielded by the classifier to develop our estimator. The top-K RUL values can be used to drive the execution of conservative or aggressive PHM strategies, or be tracked over time to develop robust RUL estimators that leverage the history of RUL estimates. The performance of the proposed RUL estimators is illustrated on a dataset from NASA's Prognostics Center of Excellence.

1. INTRODUCTION

Modern manned and unmanned vehicles are composed of complex systems that must work together to transform a fuel source into propulsion, provide navigation capabilities, and a rudimentary set of safety features, at a minimum. Any individual component malfunction within these systems may cause a cascading failure effect that could jeopardize the safety of the systems and its ability to accomplish the intended purpose. Technically, failures can be characterized as obsolescence, catastrophic and degradation. As parts pass their manufacturing end-of-life period, the lack of replacement parts forces subsystems or entire platforms to become obsolete. Sudden failures, such as the loss of a stall sensor during takeoff, have immediate catastrophic consequences. Degradation occurs when the functionality of a system becomes gradually compromised. Left unattended degradation failures can lead to catastrophic system failures. System degradation can be identified by continuous monitoring of the system state, as part of a system-wide Condition-Based Maintenance (CBM) strategy. CBM can be further extended into forecasting the future state as part of Condition-Based Maintenance Plus strategies via advanced Prognostic Health Management (PHM) applications.

Prognostics is the process of correlating and processing sensor data to estimate RUL, also known as Time-to-failure (TTF), based on the history of system states (Patil, Das, Goebel, & Pecht, 2008). Detection of current and future machinery degradation failure can be applied to predict the RUL in support of autonomous and autonomic systems. Autonomy allows missions to operate with no directions from humans and autonomicity is used to complete the mission under a self-managed operation (Sterritt, 2009). Both autonomy and autonomicity require adequate situational awareness of the system's own state to make effective mission decisions.

Traditional RUL estimation approaches have been based on statistical survival analysis, which attempts to characterize the probability of survival of a system up to a given time (Kumar & Klefsjö, 1994). This family of methods often uses a Weibull distribution, whose parameters must be estimated for a given system, to model the survival probability with its expected value as a RUL estimate (Jing & Min, 2016). ML and Artificial Intelligence are widely used in autonomous and autonomic PHM systems for RUL estimation and fault detection (van der Laan & Rose, 2011). Many different prognostics measures have been applied with varying degrees of success to determine the RUL of individual systems and whole platforms. Their operational success has been largely dependent upon the failure modes considered and the sensor capabilities available (Saxena et al., 2008).

The advent of advanced data collection and processing ca-

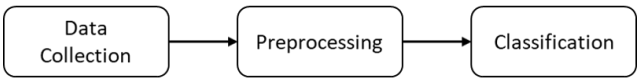


Figure 1. ML classifier flow.

pabilities in a variety of autonomous systems has motivated the development of ML methods that attempt to use time-series data for RUL estimation, see (Aggarwal et al., 2018) and references therein. Two common methods for RUL estimation based on ML are: (1) to evaluate the RUL estimation problem as a regression problem where a single output represents the predicted RUL of the system; or (2) to create a classifier where each class represents a RUL estimate for the system. The process of using ML for RUL estimation can be represented as a sequential process comprising data collection, data preprocessing, and classification. As depicted in Fig. 1 for the case when a classifier is used, the input data are comprised of measured time-series data segments. These data points are first processed to remove redundant information, reduce the noise, and extract the aggregate features useful for classification. Next, the data is also normalized and standardized to mitigate numerical instabilities that often affect the *training* of the classification block. Finally, the *preprocessed* data are moved to the classification block, yielding a classification label for the time-series data segment. Although many classification algorithms have been developed, such as decision trees, support vector machines, and K-nearest neighbors (Hastie, Tibshirani, & Friedman, 2017), using ML methods on time-series data remains a challenging problem because of the high dimensionality inherent to the time-series data themselves. Long short-term memory (LSTM) networks, a type of recurrent neural network, have recently yielded state-of-the-art results for classification and forecasting of time-series data (Vincent & de Brebisson, 2015; Sezer, Gudelek, & Ozbayoglu, 2020).

In this paper, we model the RUL estimation problem as a classification problem over a finite time-horizon. The framework developed in this paper can be applied to any classifier that yields a probability distribution over classes as its output. The classification labels correspond to a RUL estimate if a failure is predicted to occur during the time-horizon considered, or correspond to a no-failure-expected indicator if a failure is not expected. Rather than considering a single point RUL estimate, our approach uses the top-K (K-largest) probabilities to identify the top potential RUL values. Our studies reveal that a RUL estimator that always selects the RUL value associated with the class assigned the largest probability value by the classifier as its RUL estimate, i.e., using the winner-take-all rule, can yield inconsistent results over time. The top-K RUL estimator provides the top-K probability profile which can be averaged to mitigate the estimation ambiguity present in a winner-take-all approach. Lower and upper bounds RUL estimates can also be obtained from the top-K RUL estimators from the minimum and maximum operations of the RUL values. These RUL bounds are used to develop aggressive and conservative PHM policies and will play a critical role in the execution of timely and effective mitigation fault behaviors for the system.

The main contributions of this work are that: (i) it proposes an approach to RUL estimation based on a classification framework that combines multiple single-point estimates to compute a family of estimators in support of the selection of different mitigation behaviors based on the aggressiveness desired for the PHM application; (ii) it develops a family of model-comparison metrics that capture the notion of false negative and false positive counts, and allows control over the criticality from both RUL estimation and autonomy software integration perspectives; and, (iii) it proposes a method for processing RUL estimates to mitigate inconsistencies across sequential estimation periods.

The paper is organized as follows. In Section 2, the general problem is defined. Section 3 introduces a quantized RUL estimator and a family of top-K RUL estimators is developed. Section 4 discusses how to evaluate multi-classification temporal prediction. Section 5 presents the numerical results of our proposed method applied to a real data set using a LSTM classifier. Section 6 provides a method for sequential processing of RUL estimates. Finally, Section 7 concludes the paper and discusses future research directions.

2. PROBLEM FORMULATION

We consider a system with S sensors where each sensor generates a time series of sensor measurements $(x_{s,t} : t = 1, 2, \ldots)$ of arbitrary length, with $x_{s,t} \in \mathbb{R}$ denoting the measurement obtained from the s-th sensor at time index t. Measurements are taken synchronously across all sensors at a fixed sampling interval $T_p \in \mathbb{R}_+$. Let $\mathbf{x}_s := [x_{s,1}, \ldots, x_{s,T}]' \in \mathbb{R}^T$, with $T \in \mathbb{N}$, denote a vector containing the time series data measured by sensor s over a window of T samples and $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_S] \in \mathbb{R}^{T\times S}$ the system-wide measurements over the same sampling window where $(\cdot)'$ is the transpose operator. Each $\mathbf{X}$ is associated with a tuple (t_f, ν), where $t_f \in \mathbb{R}_+ \cup \{\infty\}$ denotes a *censoring* random variable and $\nu \in \mathbb{R}_+$ a reference measuring time for when $\mathbf{X}$ was measured. If a system failure occurred, t_f denotes the time at which the system failure occurred. Otherwise, $\mathbf{X}$ is said to be *censored* and t_f is set to ∞. In the latter case, a system failure is still expected to occur, but its exact occurrence time remains unknown.

The RUL for $\mathbf{X}$ is defined as

$$y = t_f - \nu \tag{1}$$

with $y \in \mathbb{R}_+ \cup \{\infty\}$. Since in practice reliable RUL predictions over long time horizons can be unreliable, we define the

censored RUL estimate over a fixed prediction time horizon T_p as

$$\gamma(y) := \begin{cases} y & y \leq T_p \\ \infty & y > T_p \end{cases} \tag{2}$$

where the choice of T_p is application-domain dependent. Its value should capture the dynamics of the degradation process of the system that precedes a system failure and the reaction time required by the system in order to trigger appropriate fault mitigation procedures. To simplify notation, we use $\bar{y} = \gamma(y)$ as a shorthand for censored RUL values.

Given a training set $\mathcal{X} := \{(\mathbf{X}_m, \bar{y}_m)\}_{m=1}^{M}$ with M training examples, where $\mathbf{X}_m$ denotes the m-th sensor data matrix and $\bar{y}_m$ its corresponding censored RUL value, our goal is to learn a mapping $h : \mathbb{R}^{T\times S} \to \{-1, 1\} \times [0, T_p] \cup \{\infty\}$. For a new $\mathbf{X}$, h identifies whether a failure would occur in the immediate time horizon defined by T_p and provide an estimate for the RUL. A classification label $\theta(\mathbf{X}; \mathcal{W}_1) \in \{-1, 1\}$ identifies impending failures with $\theta(\mathbf{X}; \mathcal{W}_1) = 1$ indicating that the RUL estimate for $\mathbf{X}$ is in the interval $[0, T_p]$ and $\theta(\mathbf{X}; \mathcal{W}_1) = -1$ that the RUL estimate for $\mathbf{X}$ is in the interval (T_p, ∞). RUL estimates are given by the function $f(\mathbf{X}; \mathcal{W}_2)$. The sets $\mathcal{W}_1, \mathcal{W}_2$ denote the learnable parameters for h.

Once $\mathcal{W}_1$ and $\mathcal{W}_2$ have been learned the censored RUL prediction for $\mathbf{X}$ is $\hat{y} = h(\mathbf{X}; \mathcal{W}_1, \mathcal{W}_2)$ where:

$$h(\mathbf{X}; \mathcal{W}_1, \mathcal{W}_2) = \begin{cases} f(\mathbf{X}; \mathcal{W}_2) & \theta(\mathbf{X}; \mathcal{W}_1) = 1 \\ \infty & \theta(\mathbf{X}; \mathcal{W}_1) = -1 \end{cases} \tag{3}$$

Characterizing h requires one to tackle a joint binary classification and regression problem. The classification problem will identify whether a failure will occur within the interval $[0, T_p]$. In the case where a failure is deemed to occur within the interval $[0, T_p]$, the regression problem estimates the corresponding RUL. Otherwise, the RUL is set to ∞ to indicate that no failure is expected to occur within the T_p prediction horizon.

Estimates $\hat{\mathcal{W}}_1, \hat{\mathcal{W}}_2$ for $\mathcal{W}_1, \mathcal{W}_2$ can be obtained as the solution of the following optimization problems:

$$\min_{\mathcal{W}_1} \sum_{m=1}^{M} L(\mathbf{1}_{\{\bar{y}_m \in [0,T_p]\}} - \mathbf{1}_{\{\bar{y}_m > T_p\}}, \theta(\mathbf{X}_m; \mathcal{W}_1)) \tag{4a}$$

$$\min_{\mathcal{W}_2} \sum_{m=1}^{M} \mathbf{1}_{\{\bar{y}_m \in [0,T_p]\}} \|\bar{y}_m - f(\mathbf{X}_m; \mathcal{W}_2)\|_2^2 \tag{4b}$$

where $L : \mathbb{R} \times \mathbb{R} \to \{0, 1\}$ is a 0-1 loss function defined as $L(a, b) = 0$, if $a = b$, and $L(a, b) = 1$, if $a \neq b$, with $a, b \in \mathbb{R}$, and $\mathbf{1}_{\{\xi\}} = 1$ if ξ is *true* and $\mathbf{1}_{\{\xi\}} = 0$ otherwise. Here, $\|\cdot\|_2$ denotes the ℓ_2-norm operator. The regression problem in Eq. (4b) uses training pairs from $\mathcal{X}$ whose $\bar{y}_m$ is not ∞.

3. A QUANTIZED RUL ESTIMATOR

In this section we introduce a joint problem formulation for identifying whether a system failure will occur over a fixed time horizon and, if so, estimating the corresponding RUL. Rather than using a two-step approach as described in Eq. (4), we formulate the RUL estimation problem as a classification problem with $N + 1$ classes, namely $\mathcal{C} := \{1, \ldots, N + 1\}$ The interval $(0, T_p]$ is divided into N subintervals. Although other interval partitioning strategies are possible, for ease of presentation we consider an equal-length, non-overlapping partitioning of $(0, T_p]$ in which the n-th subinterval in the partitioning is $((n - 1)T_p/N, nT_p/N]$. The RUL estimate associated with the n-th subinterval is defined as the largest value in the interval, i.e., $d_n = nT_p/N$. The $(N+1)$-th class identifies a situation in which a failure will not occur in the time horizon $(0, T_p]$.

In order to train a classifier, we modified the censored RUL values in $\mathcal{X}$ to generate class labels for every training example:

$$\bar{\phi}_m = \begin{cases} \sum_{n=1}^{N} n\mathbf{1}_{\{\bar{y}_m \in ((n-1)T_p/N, nT_p/N]\}} & \bar{y}_m \neq \infty \\ N + 1 & \bar{y}_m = \infty \end{cases} \tag{5}$$

Note that the sum in the first row of Eq. (5) always yields one nonzero term, which corresponds to the index of the subinterval that $\bar{y}_m$ belongs to. Given the modified training set $\mathcal{X}_C := \{(\mathbf{X}_m, \bar{\phi}_m)\}_{m=1}^{M}$, we seek to learn a mapping $\Omega : \mathbb{R}^{T\times S} \to \mathcal{P}$, where $\mathcal{P} \subset \mathbb{R}^{N+1}$ denotes the set of probability vectors $\mathbf{p} \in \mathbb{R}^{N+1}$ over the classification labels for the elements of $\mathcal{C}$, which are captured by the random variable $\bar{\Phi}$ that represents the classification label. Thus, the n-th entry of $\mathbf{p}$ corresponds to the probability of n being the correct label for $\mathbf{X}$, i.e., $p_n := \Pr(\bar{\Phi} = n|\mathbf{X}; \Omega)$ where the dependency of p_n on Ω is shown explicitly.

A maximum a posteriori (MAP) estimator can be used to choose a classification label for a new $\mathbf{X}$ as $\bar{\phi} = \arg\max_n \Pr(\bar{\Phi} = n|\mathbf{X})$. Once available, a class-label estimate $\bar{\phi} \in \mathcal{C}$ can be mapped to a RUL-value estimate via the mapping $f : \mathcal{C} \to \mathbb{R}$, which is defined as:

$$f(\bar{\phi}) = \begin{cases} d_{\bar{\phi}} & \bar{\phi} \in \mathcal{C} - \{N + 1\} \\ \infty & \bar{\phi} = N + 1 \end{cases} \tag{6}$$

Although single-point RUL estimators based on $\mathbf{p}$ can be developed, the performance of such estimators can yield inconsistent results as the number of classes considered grows large. Intuitively, slowly occurring degradation can cause the entries of $\mathbf{p}$ corresponding to neighboring RUL values to be similar. This is an artifact of the arbitrary partitioning of $(0, T_p]$ to N intervals which can yield class overlap. Thus, a small perturbation in $\mathbf{X}$ can cause its classification label to

be the one corresponding to a different class. The next section introduces a class of RUL estimators that uses the RUL values corresponding to the top-K values of $\mathbf{p}$ to mitigate the effect of inconsistent classification labeling on the RUL estimates.

3.1. RUL Estimation via a Top-K Classifier

In classification problems with a large number of classes, the traditional top-1 classification performance can yield inconsistent results. In the case of probabilistic classifiers, this behavior is reflected as one that yields multiple high-value p_n's with similar magnitude. In the context of RUL estimation, inconsistent RUL estimates can impair the ability of the system to trigger effective mitigation behaviors. Late triggering of mitigation behaviors can fail to prevent a system failure, while early triggering of mitigation behaviors can detrimentally impact the tasks being executed by the system.

In this section we propose a top-K classifier that uses the K-largest entries of $\mathbf{p}$ as a proxy to select the top-K most likely RUL values $\{d_{i_1}, \ldots, d_{i_K}\}$. Top-$K$ classifiers have been used in the context of image processing to develop robust image classifiers (Chang, Yu, & Yang, 2017). Top-K classification rules are well motivated for classifiers trained by minimizing the cross-entropy loss. In this case, it has been shown that the cross-entropy loss is top-K calibrated for any K (Lapin, Hein, & Schiele, 2016, Prop. 4). A top-K calibrated classifier will, in the limit of infinite training data, achieve the Bayes optimal top-K classification error. Other loss definitions specifically tailored for top-K classification with efficient numerical optimization characteristics can also be considered (Berrada, Zisserman, & Kumar, 2018; Lapin, Hein, & Schiele, 2015).

As noted above, this could be achieved with many different classifiers, however an LSTM-based model has been shown to perform well with extracting temporal relationships and provide effective results for RUL predictions, especially when trained using Cross Entropy Loss (Zheng, Ristovski, Farahat, & Gupta, 2017). The LSTM classifier will be used for the numerical tests in Section 5.

A fundamental question in this case is how the RUL values corresponding to the top-K classification labels should be used to construct a RUL estimator. The top-K RUL estimators d^{RUL_K} are proposed as follows:

$$d^{\text{RUL}_K}_{\text{mean}} = \sum_{k=1}^{K} w_k d_{i_k} \tag{7a}$$

$$d^{\text{RUL}_K}_{\text{min}} = \arg\min_k \{d_{i_k} : k = 1, \ldots, K\} \tag{7b}$$

$$d^{\text{RUL}_K}_{\text{max}} = \arg\max_k \{d_{i_k} : k = 1, \ldots, K\} \tag{7c}$$

where the weights $\{w_1, \ldots, w_K\}$ satisfying $w_k \in [0, 1]\ \forall k$ and $\sum_{k=1}^{K} w_k = 1$.

Algorithm 1 RUL Estimator via Top-K Classifier

Require: A mapping $\Omega : \mathbb{R}^{T\times S} \to \mathcal{P}$ and K.

1: Let $\mathbf{X} \in \mathbb{R}^{T\times S}$ denote a new sensor-data matrix.
2: Compute $\mathbf{p} = \Omega(\mathbf{X}) \in \mathbb{R}^{N+1}$.
3: Let $\{p_{i_1}, \ldots, p_{i_K}\}$ denote the top-K entries of $\mathbf{p}$ and $\{d_{i_1}, \ldots, d_{i_K}\}$ their corresponding RUL values.
4: Set $P_K = \sum_{k=1}^{K} p_{i_k}$ and compute $w_k := p_k/P_K, \forall k$.
5: Compute $d^{\text{RUL}_K}_{\text{mean}}$ via (7a).
6: **return** RUL estimate $d^{\text{RUL}_K}_{\text{mean}}$.

Equation (7a) computes a convex combination of the top-K RUL estimates. Equation (7b) defines a more aggressive RUL estimator (i.e. one that will yield the shortest RUL) when compared with Eq. (7a). Equation (7c) can be justified in the cases where the early execution of corrective behaviors may not cause a significant penalty to the system goals when compared with the occurrence of a failure, such as the in-flight failure of an engine. Equation (7c) defines a more conservative RUL estimator that is applicable when the impact of the failure on the system can be tolerated for a period of time or, in the case that an autonomous platform, when the ongoing activity is more important that platform failure. Such a conservative estimate will give the system more incentive to schedule fault-mitigation behaviors with minimal impact to the system goals. An underlying assumption is that the top-K classifications will converge over time to a single value as the failure grows more imminent.

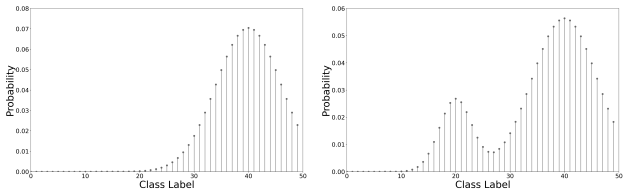
The RUL estimators in Eq. (7) can be further extended as follows:

- Setting $w_k = p_{i_k}/P_K, \forall k$, with $P_K := \sum_{k=1}^{K} p_{i_k}$.
- Estimating a failure occurrence interval $[d^{\text{RUL}_K}_{\text{min}}, d^{\text{RUL}_K}_{\text{max}}]$ via Eq. (7b) and Eq. (7c).

Algorithm 1 summarizes the proposed RUL estimation algorithm via a top-K classifier using Eq. (7a) and the first bullet above. Similar algorithms can be obtained for Eq. (7b) and Eq. (7c) after modifying Procedure 4 of Algorithm 1 appropriately. The following section discusses various metrics for assessing the performance of the estimators in Eq. (7).

3.2. Implications of the Selection of the K Value

The choice of parameter K impacts the performance of the top-K RUL estimator. The appropriate choice for K is influenced by the specific classifier design, the resolution of the RUL estimator defined by T_p/N, and the dynamics of the degradation within the time-series data. Multiple K values may be optimized using a cross-validation procedure with the evaluation metrics proposed in Section 4. For a classifier with inconsistent RUL estimates, one would expect to see an initial decrease in classification error as the value for K is increase, which is reversed after a threshold for K is crossed. The number of target classes also plays a role in the selection



(a) Unimodal PMF (b) Multimodal PMF

Figure 2. Sample class PMF's generated by a 50-class classifier. Each class index maps to a well-defined RUL value as described in Eq. (5).

of *K*. The relative importance of each class increases as the number of classes decreases. Thus, a RUL estimator using a classifier with a larger number of classes may benefit from using a larger *K*.

The probability mass function (PMF) can be used to characterize the output of the classifier. If the PMF is unimodal, as shown in Fig. 2a, all of the top probabilities will be very close to the original estimate and the improvement will be minor. If the PMF is multimodal, as shown in Fig. 2b, the benefit of using the top *K* probabilities increases.

The following methods for selecting *K* are proposed as potential starting points:

- **Static *K* Value** - K is selected via a cross-validation procedure that would follow the classifier's validating process. The RUL-estimator error can be used as an indicator to identify at what point increasing K might no longer improve the quality of the RUL estimator. An example of using a static K value is demonstrated in Section 5.

- **Dynamic *K* Value** - A dynamic K value can be used to overcome artifacts in $\mathbf{X}$ and can, thus, outperform a static choice for *K*. K can be updated using the entropy of the classifier PMF ($H := -\sum_{n=1}^{N} p_n \log(p_n)$), which characterizes the information content of the distribution, i.e., the amount of uncertainty in the outcome of a random variable from the distribution . A high (low) entropy value indicates a more (less) informative distribution and suggests the selection of a large (small) value for K.

4. EVALUATION METRICS FOR QUANTIZED RUL ESTIMATORS

Classification performance metrics such as accuracy, precision, recall and classification error can be used to assess the performance of RUL estimators proposed in Eqs. (6) and (7). These metrics can be extended to assess the performance of the top-K RUL estimators in Eq. (7) by mapping the corresponding estimate to $\mathcal{C}$ via

		Estimated	
		positive	negative
Actual	positive	**TP** True Positive	**FN** False Negative
	negative	**FP** False Positive	**TN** True Negative

Figure 3. Binary classification confusion matrix.

$$\bar{\phi}^{\text{RUL}_K} = \begin{cases} \sum_{n=1}^{N} n\mathbf{1}_{\{d^{\text{RUL}_K} \in ((n-1)T_p/N, nT_p/N]\}} & d^{\text{RUL}_K} \neq \infty \\ N+1 & d^{\text{RUL}_K} = \infty \end{cases} \tag{8}$$

where d^{RUL_K} denotes one of the estimators in Eq. (7). These metrics summarize the performance of the classifier while presuming that all classes are equally important and can be used to drive the selection of tuning parameters or the type of classifier implemented. Although valid single-point metrics, these metrics do not take into account the temporal aspect of the RUL estimation problem or the fact that failing to correctly estimate low-value RULs is more critical than failing to predict high-value RULs.

A confusion matrix captures the error distribution of the classifier per class. It can be applied to both binary and multi-class classification problems when the true classification labels are available. For a binary classification problem, the confusion matrix shows four different classification counts, namely true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) as shown in Fig. 3. A TP (TN) indicates a sample in the positive (negative) class was classified correctly, and an FP (FN) a sample in the negative (positive) class that was classified as positive (negative). The multi-class classification model of the confusion matrix can then be extrapolated as follows (Krüger, 2016), see Fig. 4. Per row $n \in \mathcal{C}$, the confusion matrix $\mathbf{E} \in \mathbb{N}^{(N+1)\times(N+1)}$ comprises a $1 \times (N+1)$ vector whose n'-th entry is $\sum_{m:c_m=n} \mathbf{1}_{\{\hat{n}_m=n'\}}$. The entries of the n-th row of $\mathbf{E}$, with the n-th entry removed, correspond to the FN count for class n. Similarly the entries of the n-th column of $\mathbf{E}$, with the n-th entry removed, correspond to the FP count for class n. Let $\mathbf{1}$ denote a vector of ones with appropriate dimensionality, $\text{diag}(\mathbf{E})$ as an $(N+1) \times (N+1)$ matrix comprising the main-diagonal entries of $\mathbf{E}$ on its main diagonal, and $(\cdot)'$ as the transpose operator. Thus, the $(N+1) \times 1$ vector $\boldsymbol{\alpha} := (\mathbf{E} - \text{diag}(\mathbf{E}))\mathbf{1}$ captures the FN count profile and the $(N+1) \times 1$ vector $\boldsymbol{\beta} := (\mathbf{E} - \text{diag}(\mathbf{E}))'\mathbf{1}$ captures the FP count profile yielded by h (as defined in Eq. (3)).

Quantized RUL estimators can be compared on the bases on these two profiles and their accuracy score $A \in [0, 1]$ through, e.g., the Euclidean distance between $\Theta := (\|\boldsymbol{\alpha}\|_2, \|\boldsymbol{\beta}\|_2, 1 -$

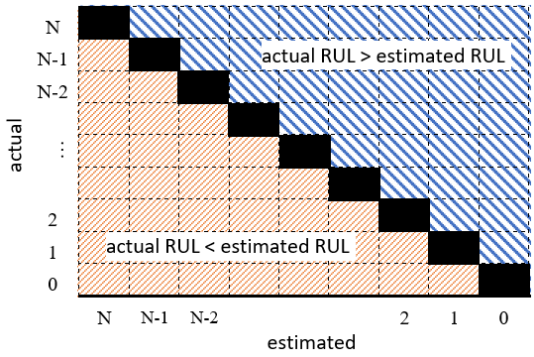


Figure 4. N-ary classification confusion matrix.

A) and the ideal score tuple $(0, 0, 0)$. This approach, however, ignores the temporal aspect of the RUL estimation problem and the fact that a false negative estimate that predicts a RUL that is smaller than true RUL is preferable to one that predicts a RUL that is larger than the true RUL. The former case would give the system a chance to react to an impending failure while the latter one would not.

In the context of RUL estimation, given a class n all FN values assigned to classes $n' > n$ should be weighed more than those assigned to classes $n'' < n$. This can be achieved for each n by using a masking function defined entry-wise as:

$$g_n(n') = \begin{cases} \lambda_1 & n' < n \\ \lambda_2 & n' \geq n \end{cases} \tag{9}$$

with scalars $0 < \lambda_1 < \lambda_2$. Let $\mathbf{G} := [\mathbf{g}_1, \ldots, \mathbf{g}_{N+1}]'$, with $\mathbf{g}_n := [g_n(1), \ldots, g_n(N+1)]'$, denote the resulting masking matrix. Then one can define an adjusted profile $\boldsymbol{\alpha}_{\text{adj}} := [\mathbf{G} \circ (\mathbf{E} - \text{diag}(\mathbf{E}))]\mathbf{1}$, where $\circ$ denotes the Hadamard product. A similar argument can be used to argue that for a given class n FPs assigned to classes indexed by n' with $n' < n$ should be weighed more since they will convey an unnecessary sense of urgency for action to system. With these observations, it is possible to define adjusted FP $\boldsymbol{\alpha}_{\text{adj}}$ and FN $\boldsymbol{\beta}_{\text{adj}}$ profiles. Then, the tuple $(\|\boldsymbol{\alpha}_{\text{adj}}\|_2, \|\boldsymbol{\beta}_{\text{adj}}\|_2, 1-A)$ can be used to assess the quality of the Quantized RUL estimator by assessing its Euclidean distance from the tuple $(0, 0, 0)$ as before.

5. NUMERICAL TESTS

In order to place this problem into a real-world context, we consider an autonomous platform monitoring a number of subsystems throughout a mission. This section illustrates the top-K RUL estimation framework proposed in this paper using a specific classifier implementation applied to the turbofan data obtained from the NASA's Prognostics Center of Excellence (PCoE) (Ramasso & Saxena, 2014).

5.1. Turbofan Dataset Description

In order to provide a numerical demonstration for the top-K RUL estimation framework developed in this work, we use the turbofan data as provided by NASA's Prognostics Center of Excellence (PCoE) (Ramasso & Saxena, 2014). This dataset was originally used for a data challenge circa 2008, and then released for public access and development of data-driven models for predictive analytics. The training data represents run-to-failure for turbofan components while the test data set is composed of partial failure trajectories. The goal of the data challenge was to identify remaining useful life at the end of each trajectory. In our case, the classifier is a deep neural network that takes the turbofan data and outputs a series of probabilities for each potential class of output.

5.2. LSTM-based Classifier Description

Inspired by the work in (Chaoub, Voisin, Cerisara, & Iung, 2021), we chose an LSTM-based classifier to process the turbofan time-series data. LSTMs are a type of recurrent neural networks that use "computational gates" with feedback connections to control the information flow across the network, and thereby to remember information at different time scales. Our LSTM-classifier comprises both LSTM cells and two multilayer perceptron (MLPs) layers. The initial MLP receives all the raw sensor data and transforms it into a feature representation for the LSTM cell. This initial MLP consists of three dense layers and learns useful representations for the normalized raw data. Hyperbolic tangent activation functions are used between each dense layer. The LSTM cell processes the data across the sequence length of the given trajectory and captures structural dependencies across the output of the first MLP block. The LSTM processed data is then passed to the second MLP, which uses hyperbolic-tangent activation functions between each dense layer but not after the output layer. The final layer of the second MLP provides an array of dimension number of sequence by number of classes from which class predictions for each sequence step in time can be extracted. The final MLP layer is extended by a softmax layer that maps the logits output of the MLP into $[0, 1]$ values, which can be interpreted as probabilities. For a given $\mathbf{X}$, the trained LSTM defines Ω and the output of its softmax layer corresponds to the $\mathbf{p}$ over the quantized RUL horizon. In the next section an LSTM is used together with Algorithm 1 to estimate the RUL for several data trajectories from the turbofan dataset.

5.3. Numerical Tests on Turbofan Dataset

The LSTM classifier described in Section 5.2 receives full engine run-to-failure trajectories as inputs and predicts the RUL at each time step. Each engine trajectory is a different length and over-sampling the minority classes destroys the time-series nature of the data. In an effort to mitigate this,

the mean sequence length of the trajectories was 206 with a standard deviation of 40. Trajectories within 206 +/- 40 are selected leaving 179 trajectories. Of these 40 were held aside for testing and the other 139 were used for training. Across all 179 trajectories, each sensor data column is min-max normalized together before splitting them up into their own trajectories. During both training and testing, only one trajectory is passed to the model at a time.

For the purpose of demonstration, only the full run-to-failure sequences, i.e., those traditionally used as training data, were considered, such that the gold RUL (the true RUL at each prediction point) is known for evaluation purposes. The goal of the classifier is to predict the RUL at each cycle of a given test sequence until failure occurs. The set of gold RUL values for a given test sequence is linearly decreasing to zero in each case. The softmax function is applied to the set of logits for each class at each sequence step, resulting in an array made up of the 252 classes (total possible predictions) and 125 cycles as the time horizon. Therefore, $T_p = 125$ is considered to be the end of the prediction horizon.

The set of top-K probabilities and the corresponding RUL values can then be extracted per time index t as shown in Fig. 5. Only one set of sample trajectories for the top-3 probabilities is shown. The performance of the RUL estimators is assessed via

$$\rho(\hat{d}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}[\hat{d}(\mathbf{X}(t)) - d(t)]^2} \tag{10}$$

where T represents the sequence length (prediction time horizon), $\hat{d}(t)$ the estimated RUL value at time t, and $d(t)$ the true RUL value at time t. Equation (10) defines the trajectory RUL-estimate root-mean-squared error (RMSE) for the estimator $\hat{d}$. What is notable about this test is that using the largest probability to choose the RUL estimate is better than using the second-largest probability. However, using the third-largest probability is the best choice when measured via the trajectory RMSE, which in this test yielded 5.496, 5.562, and 5.180 for the RUL estimates corresponding to the first, second and third probabilities, respectively.

Fig. 6 shows the improvement of the top-3 RUL estimator in Eq. (7a) over one that uses the class associated with the largest probability to estimate the RUL. Most of the RUL predictions along the trajectory were improved when the top-3 RUL estimator was used. Further, Fig. 7 shows the minimum and maximum RUL predictions at each prediction point along the trajectory. Table 1 shows the 10-best trajectory projections based on the RMSE, where 40 testing trajectories were used. The Trajectory number in Table 1 is the order of the run used in testing. The best RUL estimate $d_{\mathrm{K}}^{\mathrm{BEST}}$, defined as the RUL estimate in the set $\{d_{i_2}, d_{i_3}, d_{\mathrm{mean}}^{\mathrm{RUL}_K}, d_{\mathrm{min}}^{\mathrm{RUL}_K}, d_{\mathrm{max}}^{\mathrm{RUL}_K}\}$ yielding the best trajectory, was compared to the RMSE

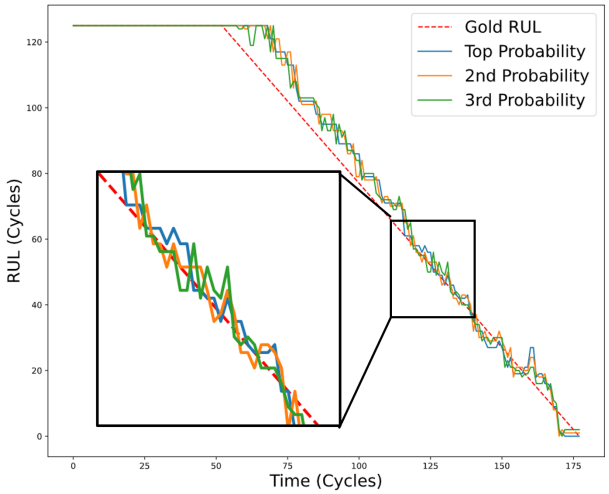


Figure 5. Top 3 probabilities for one trajectory.

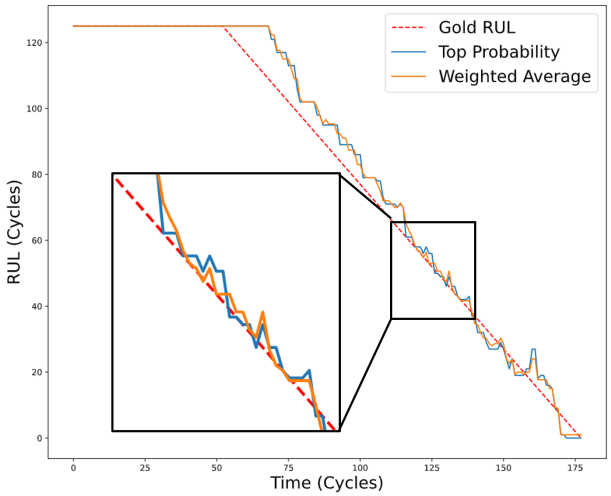


Figure 6. Top 3 probabilities averaged for one trajectory.

yielded by d_{i_1}. As the RUL estimates yielded by the top-3 probabilities are evaluated against the gold RUL, it is clear that the top probability is not always the best choice. Note that in most cases the RUL estimators in (7) yielded a better RMSE than the MAP estimate d_{i_1}.

In order to compare the models described in Section 4, two different classifier models were considered. The two models share a common architecture and training data, but use a different number of training epochs. The first model (Model 1) was trained using 95 epochs while the second model (Model 2) was trained using 35 epochs. Both models were evaluated using the six full-engine trajectories. The resulting confusion matrices are shown in Fig. 8.

The $\|\boldsymbol{\alpha}\|_2$ and $\|\boldsymbol{\beta}\|_2$ values for each model are computed and the tuples are shown in Table 2 along with their Euclidean distance from the ideal score tuple. Both the confusion ma-

Table 1. Trajectory RMSE values obtained using the RUL estimates in Eq. (7) and estimators that always choose the RUL values corresponding to the each of the first, second and third largest probabilities. Each estimator was applied sequentially to each entry of the time-series trajectory as defined in Eq. (10). The best RUL estimates obtained per trajectory are highlighted in green.

Trajectory	$\rho(d_{i_1})$	$\rho(d_{i_2})$	$\rho(d_{i_3})$	$\rho(d_{\text{mean}}^{\text{RUL}_K})$	$\rho(d_{\text{min}}^{\text{RUL}_K})$	$\rho(d_{\text{max}}^{\text{RUL}_K})$	$\rho(d_{i_1}) - \rho(d_K^{\text{BEST}})$	% Improvement
37	4.251	4.147	3.900	4.014	4.400	3.925	0.351	8.26
21	5.153	5.382	5.139	5.084	4.006	6.303	1.147	22.26
25	5.246	5.600	5.238	5.149	5.811	5.075	0.171	3.26
10	5.174	5.371	5.389	5.223	4.343	6.250	0.831	16.06
22	5.496	5.562	5.180	5.368	4.311	6.410	1.185	21.56
24	7.262	7.450	6.875	7.171	6.102	8.193	1.160	15.97
38	8.386	8.238	7.679	8.186	7.217	9.127	1.169	13.94
18	10.701	10.565	10.293	10.483	10.786	10.304	0.408	3.81
28	11.790	11.673	11.233	11.481	12.763	10.343	1.145	12.27
8	12.257	12.207	12.309	12.177	13.470	10.985	1.272	10.38

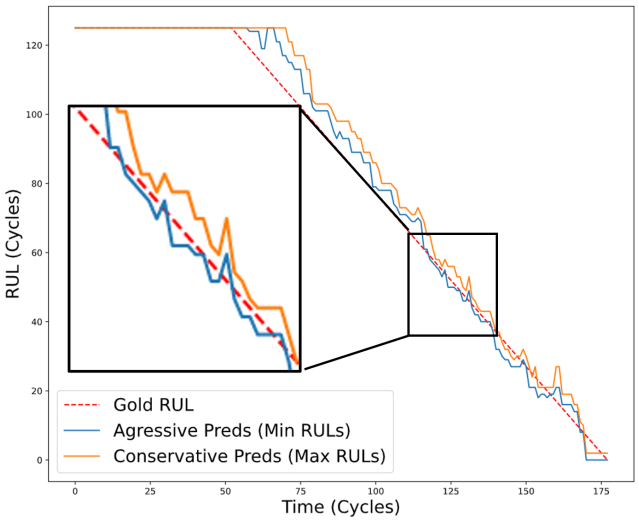


Figure 7. Minimum and maximum bounds for the RUL predictions.

trices and the Euclidean distances for each model show that Model 1 outperforms Model 2.

6. SEQUENTIAL RUL ESTIMATOR

The sequential methods incorporate prior RUL estimates to mitigate the impact of inconsistent outcomes due to the "instantaneous" noise and anomalous sensor data. Time series forecasting algorithms (TSFAs) can be used to generate history-base prediction of the RUL that can then be combined with the outcome of the top-K RUL estimator in Eq. (7a) (Suradhaniwar, Kar, Durbha, & Jagarlapudi, 2021). Common TSFAs include Kalman filters, Autoregressive processes, Moving Average processes, etc.

Table 2. Evaluation metrics obtained for Models 1 and 2.

Model	$\|\boldsymbol{\alpha}\|_2$	$\|\boldsymbol{\beta}\|_2$	1-A	$\|\Theta\|_2$
Model 1	64.078	149.853	0.933	162.981
Model 2	65.131	198.514	0.958	208.928

Given a sequence of the past RUL estimates, a Kalman predictor can be used to obtain a RUL value forecast, termed $d_{\text{Kalman}}^{\text{RUL}}$. Once the new top-$K$ RUL estimate $d_{\text{mean}}^{\text{RUL}_K}$ becomes available, the sequential RUL estimate can be computed as

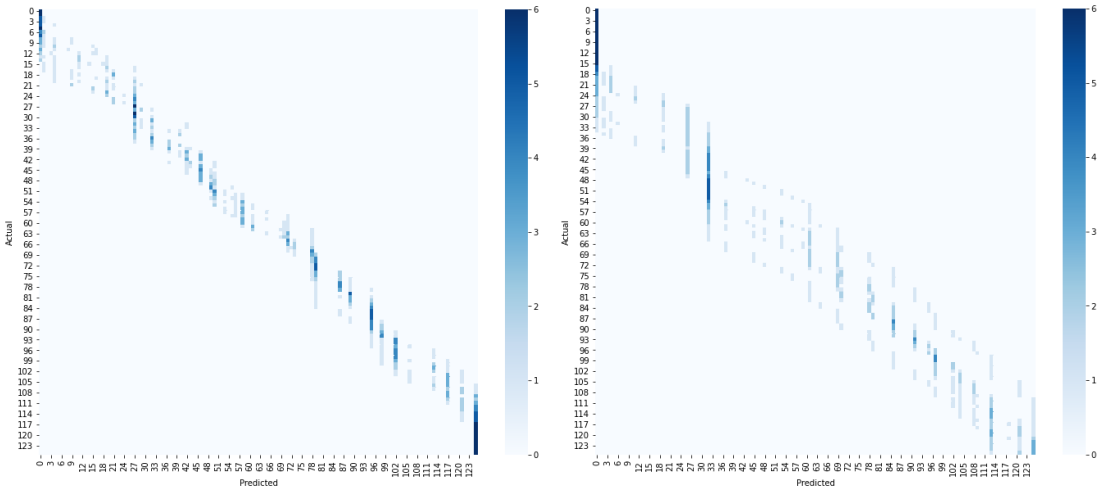
$$d_{\text{TSFA}}^{\text{RUL}} = \zeta d_{\text{Kalman}}^{\text{RUL}} + (1-\zeta) d_{\text{mean}}^{\text{RUL}_K} \tag{11}$$

with $\zeta \in [0,1]$ being a tuning parameter that adjusts the emphasis placed on the top-K and Kalman estimates. In practice, we consider a sliding window of size Φ containing the historical top-K RUL estimates for the previous Φ decision epochs, namely $\{d_{\text{mean}}^{\text{RUL}_K(\tau_{\Phi-1})}, \ldots, d_{\text{mean}}^{\text{RUL}_K(\tau_0)}\}$ where $d^{\text{RUL}_K(\tau_i)}$ denotes top-K RUL estimate at decision epoch τ_i, $i \in \{0, \ldots, \Phi-1\}$. A conceptual example of this approach is shown in Fig. 9.

Other methods, such as a Vandermonde polynomial extrapolation, can yield a RUL prediction by fitting a polynomial to a set of past RUL estimates to extrapolate the future RUL value. Methods using extrapolations make fewer assumptions on the dynamics and distribution of the data, but may require a larger set of RUL estimates for training. A demonstration of the sequential RUL estimator described in this section is outside the scope of this paper.

7. CONCLUSION AND FUTURE WORK

This paper proposed methods to account for the drawbacks of the traditional classifiers used for RUL estimation. The RUL of the platform is either considered to be ∞, in the case of no detectable degradation or degradation over a time considered too long to be accurate, or the time to a future failure. This RUL estimation problem was cast as a general classification problem for which a MAP estimator can be developed. Although this estimator can yield acceptable results, it



(a) Trained for 95 epochs

(b) Trained for 35 epochs

Figure 8. Confusion matrices for Model 1 and Model 2.

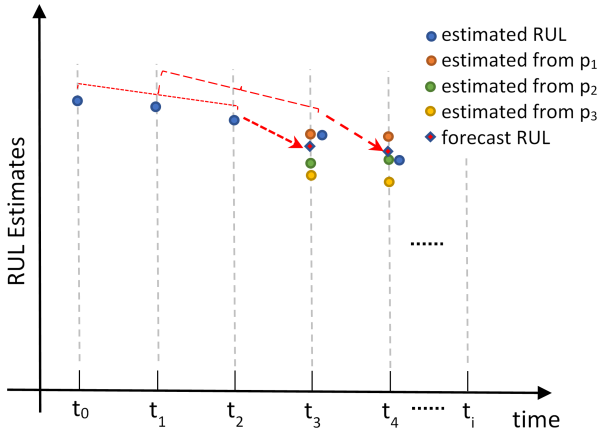


Figure 9. Sequential RUL estimator example with $w = 3$.

becomes sensitive to small perturbations and outliers as the number of classes considered by the classifier increases. As a way to mitigate this problem, a method that considers the top-K probabilities instead of just the largest one to estimate RUL was proposed. The value for K may be fixed, or variable based on the dynamics of the system. Three different top-K estimators were proposed. The weighted average estimator yielded the better estimation in terms of RMSE. The minimum-value estimator supported mission critical assessments (e.g., platform safety) such that repairs can be accomplished prior to failure. The maximum-value estimator placed higher emphasis on the completion of mission objectives (i.e., trying to accomplish as much as possible prior to the execution of an appropriate fault mitigation behavior).

Next, an approach for assessing and comparing RUL estimators based on a confusion matrix was developed. Typical metrics such as accuracy, precision, and recall work only as long as the classifier output is perfectly correlated to the true RUL. In real-world cases, two models may be vastly different in terms of RUL estimation, with one only a single minute off at a given time estimate and the other an hour off, yet both could score similarly on accuracy, precision, and recall. Combined with a custom masking function that serves to penalize late predictions (those that would occur after a platform failure), a metric for comparing RUL estimators was proposed. Finally, examples of the proposed methods were evaluated against the well-known Turbofan dataset from NASA's PCoE to demonstrate the benefits of the top-K RUL estimator.

Future work is expected to develop heuristics for dynamic selection of the top-K values based on their proximity to the top values. Numerical evaluation of Kalman prediction for RUL prediction and tracking, and dynamic selection of K are both areas that could yield additional benefits for improving predictive analytics. Additionally, we plan to consider classifiers whose objective function during training captures the fact that a subsequent top-K decision rule is used for RUL estimation.

ACKNOWLEDGMENT

This work was made possible through the collaboration between Trident Systems Incorporated and the Naval Information Warfare Systems Center Pacific under the Navy Cooperative Research and Development Agreement NCRADA-NIWCPacific-21-389.

NOMENCLATURE

CBM	Condition-Based Maintenance
FN	False Negative
FP	False Positive
$LSTM$	Long Short Term Memory
MAP	Maximum A Posteriori
ML	Machine Learning
MLP	Multilayer Perceptron
$NASA$	National Aeronautics and Space Administration
$PCoE$	Prognostics Center of Excellence
PMF	Probability Mass Function
PHM	Prognostics and Health Management
$RMSE$	Root Mean-Squared Error
RUL	Remaining Useful Life
TN	True Negative
TP	True Positive
TTF	Time to Failure

REFERENCES

Aggarwal, K., Atan, O., Farahat, A. K., Zhang, C., Ristovski, K., & Gupta, C. (2018, Dec. 12-13, Seattle, WA, USA). Two birds with one network: Unifying failure event prediction and time-to-failure modeling. In *Proc. of IEEE international conference on big data* (p. 1308-1317).

Berrada, L., Zisserman, A., & Kumar, M. P. (2018, Apr. 30 - May 3, Vancouver, BC, Canada). Smooth loss functions for deep top-k classification. In *Proc. of 6th international conference on learning representations, ICLR 2018.* OpenReview.net.

Chang, X., Yu, Y.-L., & Yang, Y. (2017, Aug. 13-17, Halifax, NS, Canada). Robust top-k multiclass SVM for visual category recognition. In *Proc. of acm conference on knowledge discovery and data mining – kdd 2017* (p. 75 - 83). New York, NY, USA: Association for Computing Machinery.

Chaoub, A., Voisin, A., Cerisara, C., & Iung, B. (2021). Learning representations with end-to-end models for improved remaining useful life prognostics. *CoRR*, *abs/2104.05049*. Retrieved from `https://arxiv.org/abs/2104.05049`

Hastie, T., Tibshirani, R., & Friedman, J. (2017). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer.

Jing, R., & Min, S. (2016). Prediction method study on the remaining useful life of plant new varieties rights based on Weibull survival function and Gaussian model?taking hybrid rice variety for example. *Agricultural Science and Technology*, *17*(4).

Krüger, F. (2016). *Activity, context, and plan recognition with computational causal behaviour models* (Unpublished doctoral dissertation). Universitat Rostock.

Kumar, D., & Klefsjö, B. (1994). Proportional hazards model: a review. *Reliability Engineering and System Safety*, *44*(2).

Lapin, M., Hein, M., & Schiele, B. (2015, Dec. 7-10, Montreal, Canada). Top-k multiclass SVM. In *Proc. of the 28th international conference on neural information processing systems - volume 1* (p. 325 - 333). Cambridge, MA, USA: MIT Press.

Lapin, M., Hein, M., & Schiele, B. (2016, Jun. 27-30, Las Vegas, NV, USA). Loss functions for top-k error: Analysis and insights. In *Proc. of IEEE conference on computer vision and pattern recognition, CVPR* (pp. 1468–1477). IEEE Computer Society.

Patil, N., Das, D., Goebel, K., & Pecht, M. (2008, Oct. 6-8, Denver, CO, USA). Identification of failure precursors for insulated gate bipolar transistors. In *Proc. of the first international conference on prognostics and health management* (p. 1-5).

Ramasso, E., & Saxena, A. (2014). Performance benchmarking and analysis of prognostic methods for CMAPSS datasets. *International Journal of Prognostics and Health Management*, *14*, 1-15.

Saxena, A., Celaya, J., Balaban, E., Goebel, K., Saha, B., Saha, S., & Schwabacher, W. (2008, Oct. 6-8, Denver, CO, USA). Methods for evaluating performance of prognostic techniques. In *Proc. of the first international conference on prognostics and health management* (p. 1-17).

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning : A systematic literature review: 2005?2019. *Applied Soft Computing*, *90*, 106181.

Sterritt, R. (2009, Oct. 13-14, Skövde, Sweden.). Autonomous and autonomic systems: Paradigm for engineering effective software-based systems? In *Proc. of 33rd annual IEEE software engineering workshop* (p. 57-57). doi: 10.1109/SEW.2009.22

Suradhaniwar, S., Kar, S., Durbha, S., & Jagarlapudi, A. (2021). Financial time series forecasting with deep learning : A systematic literature review: 2005?2019. *Sensors*, *21*.

van der Laan, M. J., & Rose, S. (2011). *Targeted learning: Causal inference for observational and experimental data* (1st ed.). Springer.

Vincent, P., & de Brebisson, X., A.and Bouthillier. (2015). Efficient exact gradient update for training deep networks with very large sparse targets. *NeurIPS*, *15*(6), 1108-1116.

Zheng, S., Ristovski, K., Farahat, A., & Gupta, C. (2017, Jun. 19-21, Dallas, TX, USA). Long short-term memory network for remaining useful life estimation. In *Proc. of the IEEE international conference on prognostics and health management (ICPHM)* (p. 88-95).