Benchmark proposal: Neural Network Based Remaining Useful Life Predictor

Executive summary: Collins Aerospace proposes a benchmark problem for the 2022 Competition on the Verification of Neural Networks (VNN). Current document contains problem background and the description of models and formal properties that are provided to the participants.

1. Background: Remaining Useful Life

Remaining Useful Life (RUL) is a widely used metric in Prognostics and Health Management (PHM) that manifests the remaining lifetime of a component (e.g., mechanical bearing, hydraulic pump, aircraft engine). RUL is used for Condition-Based Maintenance (CBM) to support aircraft maintenance and flight preparation. It contributes to such tasks as augmented manual inspection of components and scheduling of maintenance cycles for components, such as repair or replacement, thus moving from preventive maintenance to *predictive* maintenance (do maintenance only when needed, based on component's current condition and estimated future condition). This could allow to eliminate or to extend service operations and inspection periods, optimize component servicing (e.g., lubricant replacement), generate inspection and maintenance schedules, and obtain significant cost savings. RUL could also highlight areas for inspection during the next planned maintenance, i.e., it could be used to move up a maintenance/inspection action to prevent component failure. Finally, RUL function can also be used in airborne (in-flight) applications to dynamically inform pilots on the health state of aircraft components during flight.

Multivariate time series data is often used as RUL function input, for example, measurements from a set of sensors monitoring the component state, taken at several subsequent time steps (within a time window). Additional inputs may include information about current flight phase, mission and environment. Such highly multi-dimensional input space motivates the use of Deep Learning (DL) solutions with their capabilities of performing automatic feature extraction from raw data. A number of solutions based on Deep Neural Networks (DNNs) has emerged over recent years (e.g., [1] [2] [3]).

An example of a DL-based RUL function is illustrated on **Figure 1** in the context of predicting the RUL of a mechanical bearing component, which is a part of a drivetrain. Machine Learning (ML) Constituent incorporates both the DL model and pre/post processing software. It accepts as inputs a number of *condition indicators*, which are statistical indicators computed from sensor measurements, as well as information about current flight (regime, mission, environment) provided by avionics software. The output of ML Constituent, i.e., the RUL, can be used both by the pilot in-flight and by the maintenance team on ground, depending on the application.

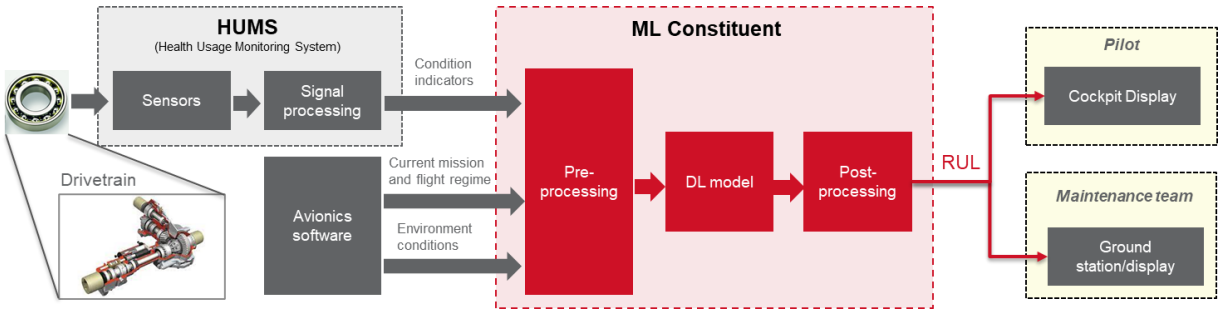

***Figure 1**. High-level overview of the ML Constituent for RUL estimation, and its operating environment*

2. Benchmark: CNN Model

We propose a Convolutional Neural Network (CNN) model adapted from [4] as a benchmark for the competition. The CNN accepts as input a sequence (time window) of inputs over several time steps. The inputs are snapshots of condition indicators at a given time step (as described above), as well as other metrics. A number of convolutional layers are used to apply 1D convolutions to the inputs along the time sequence direction. Extracted features are then merged together via a fully connected layer. Dropout is used to mitigate overfitting. Activation functions at all layers are Rectified Linear Units (ReLUs). The CNN performs a *regression* task and outputs a numerical value, which is the RUL.

Several neural networks of different complexity are provided (in ONNX format):

- **NN_rul_small_window20.onnx** (number of ReLUs - 5500)
- **NN_rul_full_window20.onnx** (number of ReLUs - 10300)
- **NN_rul_full_window40.onnx** (number of ReLUs - 28300)

All networks have been trained using the same dataset. The motivation for providing several networks is the scalability study. The number of ReLUs is different for each network. This seems to be one of the key complexity metrics for many VNN tools. Internally, networks have the same architecture/layers, but some different hyperparameters, such as the number of filters in convolutional layers ("small" networks have fewer filters, "full" networks have more). Also, two window sizes (20 and 40) have been used to generate the networks. While the number of input features remains constant for all networks, manipulating the window size allows to change the input space size (2x in the case of window sizes used). Change in window size has a more significant impact on the overall CNN complexity.

3. Benchmark: Properties

We propose several classes of properties for the NN-based RUL estimation function. First two classes (robustness and monotonicity) are local, i.e., defined around a given point. To address the requirement for randomizing the inputs to VNN tools, we provide a script with adjustable random seed that can generate these properties around input points randomly picked from a test dataset. Properties of the last class ("if-then") are defined over input ranges. We have generated a list of such properties and provide means for randomly selecting properties from this list.

Below, a short description of each property class is provided.

3.1. Robustness properties

These are *local* robustness properties defined in the "delta-epsilon" form:

$$\|x' - x\| < \delta \Rightarrow \left\|\hat{f}(x') - \hat{f}(x)\right\| < \varepsilon, \text{ where } x, x' \in X \text{ and } \delta, \varepsilon \in R_{>0}$$

The property requires that for any input perturbation bounded by δ the output must not deviate by more than ε (infinity norm is used to define perturbations). Provided robustness properties differ in the number of perturbed inputs and perturbation magnitude. Perturbations are applied only on *condition indicator* inputs (first 8 columns of the input sequence), at random.

3.2. Monotonicity properties

These *local* properties require that the value of the RUL decreases monotonically, with the increase in the values of certain input features. For example, the forward decreasing monotonicity property can be expressed as follows:

$$\forall\, x' : x \;\leq\; x' \leq x + \delta \Rightarrow f(x') \leq f(x) + \varepsilon$$

The property requires that for each input x' that is larger than the selected input x by up to δ the output $f(x')$ of the NN must be smaller than the output $f(x)$ except for small admissible non-monotonicity ε. The latter means that the output at x' the function *may* be non-monotonic, but this is bounded. This is further illustrated on **Figure 2**, where two non-monotonic regions are present. While in the second one the non-monotonicity is admissible (the predicted RUL output is slightly larger), in the first one the function increase exceeds ε, which is a property violation.

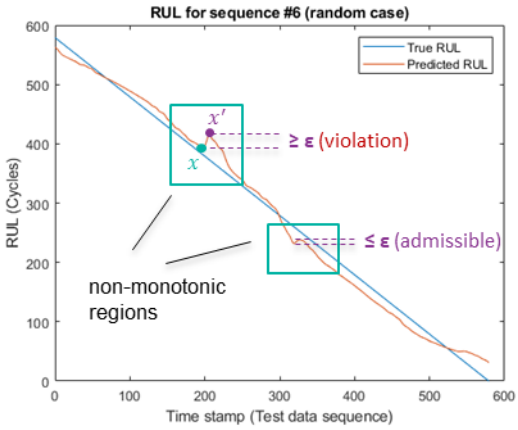

***Figure 2**. Illustration of local monotonicity property for the RUL.*

For the current benchmark, monotonicity properties are generated by randomly increasing certain inputs. With the increased values of these inputs the RUL is normally expected to decrease. An input is chosen randomly, the increase is applied for this input in the entire time window (at all steps).

3.3. If-Then properties

These properties are formulated as follows: IF the CNN inputs are in given ranges, THEN the output (RUL) must be in an expected range. To generate such properties, input ranges of some inputs have been broken down into several sub-ranges (e.g., Low, Medium, High) with corresponding lower and upper bounds. For each combination of these sub-ranges, an expected output range has been estimated. Given these numerical bounds on input/output ranges, if-then properties are straightforward to formulate.

NOTE: given the number of combinations of input ranges, the number of if-then properties can be extremely large. Therefore, for the competition we have generated only a small subset of such properties with different complexity, based on size and the number of the ranges. The properties are randomly selected from this subset. **If-then properties are currently the hardest to verify.**

4. Closing Remarks

Property verification of the RUL CNN is an important step towards *certification* of such ML-based functions. European Aviation Safety Agency (EASA) in their First Usable Guidance for Level 1 Machine Learning Applications [5] emphasizes Formal Methods as anticipated means of compliance for the verification of robustness of ML models.

References

[1] X. Li, Q. Ding and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety,* pp. 1-11, 2018.

[2] L. Ren, J. Cui, Y. Sun and X. Cheng, "Multi-bearing remaining useful life collaborative prediction: A deep learning approach," *Journal of Manufacturing Systems,* vol. 43, pp. 248-256, 2017.

[3] M. Yuan, W. Yuting and L. Li, "Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network," in *IEEE international conference on aircraft utility systems (AUS)*, 2016.

[4] "Remaining Useful Life Estimation using Convolutional Neural Network," MathWorks, 2021. [Online]. Available: https://www.mathworks.com/help/releases/R2021a/predmaint/ug/remaining-useful-life-estimation-using-convolutional-neural-network.html.

[5] EASA, "Concept Paper: First usable guidance for Level 1 machine learning applications," EASA, 2021.