International Journal of Engineering Innovation and Computational Science

Adaptive Neural Network Architectures for Predictive Maintenance and Remaining Useful Life Estimation in Smart Manufacturing

Helge S. Stein [1*], **Pascal Friederich** [2]
[1] Professor, Materials Informatics, Technical University of Munich (TUM), Munich, Germany
[2] Associate Professor, AI for Materials Science (AiMat), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
*** Corresponding Author:** helgs.stein@tum.de

Citation: H. S. Stein and P. Friederich, "Adaptive Neural Network Architectures for Predictive Maintenance and Remaining Useful Life Estimation in Smart Manufacturing," *International Journal of Engineering Innovation and Computational Science*, vol. 1, no. 1, pp. 62–78, 2026.

Received: 13 Dec 2025
Accepted: 14 Mar 2026

Abstract

Adaptive prognostic models remain difficult to trust when operating conditions and sensor quality change together. This study examines a compact neural architecture for remaining useful life estimation that separates conditional computation from supervised model updating. Three neural experts process causal sensor summaries over 8, 16, and 32 cycles, while a context gate assigns their weights from load and measurement availability. A restricted update changes only the gate and expert output layers, and independent split conformal calibration supplies prediction intervals. The architecture is evaluated through an executed, reproducible synthetic experiment comprising five independently generated fleets, each containing 900 assets. Four configurations isolate the effects of gating and updating; no industrial measurements or public benchmark results are presented as experimental evidence. Under the simulated target shift, the updated gated model achieves a mean root mean squared error of 24.23 cycles, compared with 31.94 for its frozen counterpart and 25.29 for the updated uniform-weight configuration. Its 4.2% advantage over the latter is modest and not consistent across all repetitions. Nominal 90% intervals attain 90.9% mean target coverage, but coverage falls to 73.6% after additional sensor corruption. The uniform-weight updated model also performs better under this stress. Only 174 of 2,550 parameters require updating, although adaptation and calibration together still consume 160 labelled target assets. These results support controlled adaptation as a useful engineering option while exposing its dependence on representative labels and calibration conditions. The contribution is a transparent architectural experiment and validation protocol, rather than evidence of deployment readiness or superiority over established prognostic models.

Keywords: Predictive Maintenance, Remaining Useful Life, Adaptive Neural Networks, Operating Conditions, Uncertainty Calibration.

INTRODUCTION

Predictive maintenance becomes useful when it changes a decision at the right time. A production planner needs to know whether a spindle can complete another batch, whether a bearing should wait for a scheduled shutdown, or whether a changing vibration pattern requires inspection now. A low average prediction error helps, but it does not settle any of these questions. The answer also depends on operating load, the consequences of premature replacement, available maintenance capacity, and how much confidence the prediction deserves. Reviews of deep learning for maintenance have accordingly treated model selection as part of a wider monitoring and intervention process [1]. Remaining useful life, or RUL, expresses the time between the present observation and a specified loss of acceptable function. That last qualification matters. A dimensional tolerance violation, a vibration threshold, and a catastrophic fracture define different endpoints. The established prognostics literature connects data acquisition, health assessment, degradation modelling, and life prediction because errors introduced early in that chain survive sophisticated downstream modelling [2]. A recurrent temptation in this field is to spend more effort refining the estimator than defining what failure actually means. An elaborate network cannot repair an ambiguous endpoint.

Smart manufacturing adds another complication. Equipment does not operate under a single stable

Copyright © 2026 by Author/s. IJEICS is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

recipe. Product changes alter duty cycles; process adjustments move temperature and torque; sensors are replaced, recalibrated, or temporarily disconnected. The observed signal combines degradation with these other influences. A network trained on one mixture can perform poorly when that mixture changes, even if the underlying component and failure mechanism remain familiar. The N-CMAPSS dataset illustrates the value of incorporating realistic operating profiles into simulated prognostics research, while still requiring a clear distinction between simulation and physical fleet evidence [3].

Adaptation offers several responses, but the term often obscures what actually changes. Conditional computation changes the relative contribution of existing network components for each observation. Parameter adaptation changes learned weights using new data. Calibration changes the mapping from prediction errors to uncertainty intervals. These operations have different information requirements and failure modes. Hou et al. demonstrate why incomplete target degradation trajectories are particularly troublesome for domain adaptation: matching distributions can align different stages of deterioration rather than equivalent health states [4]. More flexibility does not remove that identification problem.

Recent work on adaptive mixtures of experts and multiscale feature fusion provides a plausible basis for condition-dependent processing [5]. A short window can respond quickly to a load transition, whereas a longer window may stabilize an estimate of gradual wear. Yet a learned gate can also exploit a training-specific association between operating regime and expected lifetime. Once that association changes, the supposedly adaptive component becomes a source of error. The relevant question is therefore narrower than whether adaptation improves accuracy: which component helps, under which shift, and with what additional information?

This paper investigates that question using one compact architecture, a context-gated multiscale neural regressor. Its experts operate on transparent temporal summaries, and its gate receives operating context rather than future degradation information. A restricted supervised update changes a small subset of parameters. Independent residual calibration then attaches an interval to each estimate. Figure 1 shows the complete arrangement. The design deliberately exposes the boundaries between sensing, prediction, updating, and calibration so that a favourable result cannot be attributed vaguely to an adaptive framework.

The empirical contribution is an executed synthetic study, not a report of industrial deployment. Five independent repetitions compare uniform and learned expert weights, each with and without a target-domain update. A paired stress condition adds missing readings and sensor offsets without changing the latent failure times. This permits a direct examination of conditional computation, update value, source-domain retention, and interval reliability. The study does not establish superiority to convolutional networks, recurrent networks, transformers, or physics-based prognostic models. Those comparisons require shared datasets and a substantially broader experimental programme. The main result is deliberately qualified. Updating improves target prediction, but the advantage attributable to gating is small, and the gate does not protect the model against additional sensor corruption. That outcome is useful. It identifies where complexity earns its place and where a simpler configuration remains competitive. The paper develops the method, reports the numerical evidence, and connects these limits to the decisions that a maintenance system would eventually have to support.

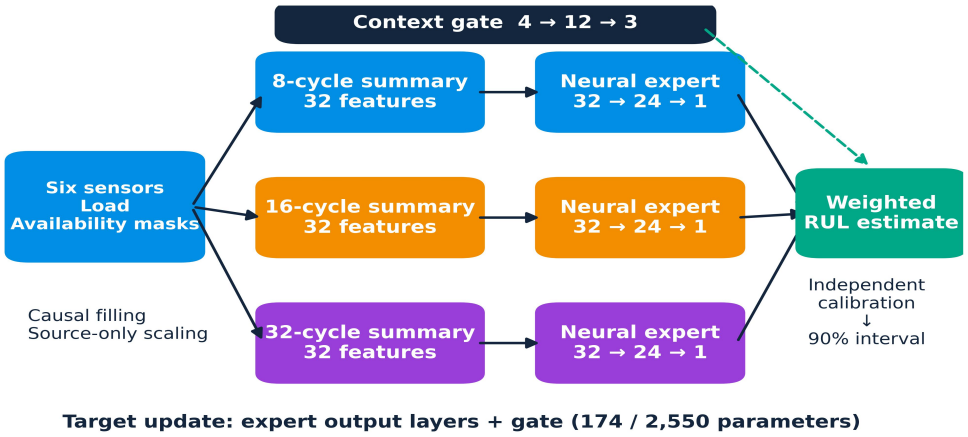


Figure 1. Overview of Multiscale Temporal Experts, Context-Gated RUL Fusion, Residual Calibration, and Restricted Parameter Updates (174/2,550 active).

LITERATURE REVIEW

Learning Degradation from Sensor Histories

Neural prognostics has progressed from learning compact health indicators toward estimating RUL directly from multivariate histories. The convolutional approach of Li, Ding, and Sun is a foundational example of extracting degradation information from sensor windows [6]. Its continuing relevance lies less in any particular architecture than in the treatment of local temporal structure as learnable information. Raw measurements rarely provide a unique health state at a single instant. Their development over time can help distinguish persistent deterioration from a temporary operating excursion.

Multiscale models extend this reasoning. Xu, Zhang, and Miao combine temporal convolution with attention to extract and integrate information at different scales [7]. The engineering motivation is sound: a signal can contain fast transients and slow wear trends simultaneously. However, increasing the number of temporal branches also expands the opportunity to fit incidental correlations. Window length, sampling rate, and operating transitions interact. A comparison made after changing all three cannot isolate the contribution of a new attention mechanism. Here, temporal summaries are held fixed across the experimental configurations so that the gating contrast remains identifiable.

Physics-informed work addresses a different weakness. Li et al. distinguish several ways of combining mechanistic knowledge with data-driven prediction and discuss the limits of unconstrained models [8]. Physical structure can make extrapolation more defensible, but only when the assumed mechanism applies. A bearing with lubrication failure is not necessarily described by a wear relation calibrated under clean, steady operation. The present simulator uses an explicit damage process to make the experimental conditions inspectable. The predictor never receives that damage state or equation, so it should not be described as a physics-informed neural network.

Conditional Computation and Adaptive Architectures

The mixture-of-experts principle is well established. Jacobs et al. introduced adaptive combinations of local experts in which a gating mechanism allocates responsibility among component models [9]. For prognostics, this offers an attractive division of labour: different experts may represent distinct operating contexts, temporal scales, or degradation patterns. These interpretations are hypotheses about learned behaviour, however. A large gate weight identifies computational influence; it does not establish that the selected expert represents a particular physical failure mechanism.

Zhou and Wang apply adaptive feature fusion and expert mixtures to joint health-status assessment and RUL prediction [5]. Their formulation also addresses relationships between tasks, a feature absent from the single-output architecture studied here. That distinction prevents an inflated novelty claim. The proposed

method introduces neither expert mixtures nor multiscale representation. Its purpose is to examine their value within a restricted, auditable update protocol. Three small experts are sufficient for that question. Adding more branches would change the model capacity before the underlying adaptation argument had been tested.

Conditional computation also differs from neural architecture search or structural growth. The network in this study retains the same number of layers, experts, and parameters throughout operation. Only the input-dependent expert weights vary during ordinary inference. A separate supervised procedure can revise selected parameters after labelled target histories become available. Calling both processes adaptive is reasonable only if their roles remain explicit. Otherwise, a reported improvement may quietly combine extra data, extra training, and extra representational capacity.

Transfer Across Operating Conditions

Domain adaptation studies increasingly recognize that operating conditions are structured rather than interchangeable nuisance variables. Nejjar et al. align operation phases separately to avoid mismatches caused by unequal phase representation [10]. Lei et al. explicitly model the influence of time-varying operating conditions before domain alignment [11]. These approaches identify an important weakness of global feature matching: two overall distributions can look similar while pairing observations with different prognostic meaning. Operating context should therefore inform both the representation and the evaluation design.

Feature disentanglement offers another route. Yan et al. separate shared information from domain-specific features while coupling this representation with RUL prediction [12]. Lv et al. use source selection and target pseudo-class information to address transfer quality [13]. Such strategies can reduce dependence on target failure labels, but they introduce assumptions about which structure should transfer. Pseudo-labels are especially delicate in prognostics because an erroneous life estimate can become its own training target. This study avoids that feedback by using observed synthetic failure times for updating. It makes no claim of unsupervised adaptation.

The broader few-shot literature also makes clear that limited-label prediction is not simply ordinary supervised learning with a smaller batch [14]. Prior knowledge, task construction, and available degradation stages determine what can be learned. Thirty target assets supply the update gradients here, but another ten select the checkpoint and 120 calibrate intervals. Reporting only the first number would misrepresent the information budget. One practical lesson is easy to overlook: data that arrive every second can still provide very little new information about failure. The independent unit is usually the asset or degradation episode, not the sensor row.

Uncertainty and the Meaning of a Reliable Interval

Uncertainty in RUL arises from several sources, including future loading, unobserved material variation, sensor noise, and imperfect model specification. Hüllermeier and Waegeman explain why aleatoric and epistemic uncertainty require careful conceptual separation [15]. In a manufacturing application, the distinction may also depend on what information is available. Uncertain future load could become less uncertain when a production schedule is known. An interval that summarizes historical prediction errors does not automatically identify or separate these sources.

Bayesian neural methods provide one approach to uncertainty estimation. Benker et al. evaluate Bayesian networks trained with Hamiltonian Monte Carlo and variational inference for simulated engine RUL [16]. Yang et al. combine an LSTM-based bearing predictor with uncertainty quantification through a modified dropout procedure [17]. These studies establish that uncertainty can be treated as a model output rather than an afterthought. They do not imply that every probabilistic output is calibrated for a new factory, sensor configuration, or maintenance policy. Reliability remains an empirical and assumption-dependent property.

Conformal prediction offers a complementary calibration layer. Javanmardi and Hüllermeier apply conformal intervals to RUL estimation [18], while Romano, Patterson, and Candès develop conformalized quantile regression to accommodate varying error scale [19]. Piao, Huang, and Tsung further connect conformalized RUL estimation with uncertainty-aware interval construction [20]. The present study uses the

simpler absolute-residual form of split conformal prediction. Its constant half-width makes the limitations visible and avoids attributing improved intervals to an additional learned uncertainty model.

The guarantee requires careful wording. As explained by Angelopoulos and Bates, split conformal coverage is marginal over new observations under the relevant exchangeability assumptions [21]. It is not a guarantee for each individual machine or every stage of its life. Barber et al. examine departures from exchangeability, and Gibbs and Candès study adaptive conformal inference under distribution shift [22], [23]. Neither result licenses unchanged calibration under arbitrary sensor drift. In prognostics, delayed failure outcomes also limit how quickly a sequential calibration procedure can receive useful feedback.

Positioning the Present Study

Recent research connects uncertainty to data acquisition and measurement quality. CoPAL uses conformal uncertainty in active selection for RUL learning [24]. Wang et al. address missing and faulty sensor observations within a dedicated uncertainty-quantification framework [25]. These contributions motivate a practical evaluation question: does an interval remain useful when the information entering the predictor changes? Reporting accuracy on a clean target set alone cannot answer it.

The present contribution is a controlled study of an existing architectural principle under explicit information constraints. It combines transparent temporal summaries, context-dependent expert weighting, a restricted supervised update, and independent residual calibration into one prognostic method. The experimental contrasts remain within that method. They quantify what gating adds beyond updating and show how both respond to a sensor disturbance excluded from calibration. This scope is narrower than a comprehensive benchmarking study, but it supports a clearer account of which conclusions the evidence can bear.

METHODOLOGY

Study Design and Prediction Target

The study uses one computational method: a context-gated multiscale neural regressor with restricted target updating and split conformal calibration. A synthetic fleet supplies fully observed degradation histories so that operating shift and sensor corruption can be controlled independently. For asset i, failure occurs at the first cycle T_i when latent damage reaches one. At observation cycle t, the target is the uncapped remaining life $T_i - t$. Predictions use measurements available through t only. The damage threshold and failure time generate labels; neither enters the predictor as an input.

Five repetitions regenerate every fleet and reinitialize the networks. Each repetition contains 900 independent assets, partitioned before feature extraction. Training and validation can use several windows from an asset, but calibration and endpoint testing each use one randomly selected eligible endpoint per asset. Eligible endpoints start at cycle 31, ensuring a complete 32-cycle observation window. Sampling is uniform within each asset's observed lifetime, which defines the evaluation population explicitly. It does not reproduce the age distribution of an operating factory fleet or a technician's selection of inspection times.

Table 1 specifies the allocation. Source training uses 240 assets and source checkpoint selection uses 60. The target update uses 30 assets, with ten additional assets for checkpoint selection. Separate source and target calibration sets each contain 120 assets. The respective test sets contain 160 assets each. Additional sensor stress reuses the target test assets and endpoints, preserving their true failure times. Across five repetitions, there are 4,500 unique assets, 800 target test endpoints, and 800 paired stressed observations. No calibration or test labels enter gradient updates or checkpoint selection.

Table 1. Asset Partitions and Observation Windows in each Synthetic Repetition

Partition	Assets	Windows per asset	Purpose

Partition	Assets	Windows per asset	Purpose
Source training	240	8	Fit neural weights
Source validation	60	4	Select source checkpoint
Source calibration	120	1	Calibrate source intervals
Source test	160	1	Evaluate source retention
Target update training	30	8	Restricted supervised update
Target validation	10	4	Select update checkpoint
Target calibration	120	1	Calibrate target intervals
Target test	160	1	Evaluate target predictions

Note:Total: 900 unique assets per repetition. The stressed evaluation reuses the 160 target test assets and their endpoints. Five repetitions generate 4,500 unique assets.

Synthetic Degradation and Sensor Generation

The simulator represents a generic wear-dominated component. Each asset has an initial damage value drawn uniformly from zero to 0.03 and a wear coefficient drawn uniformly from 0.0018 to 0.0032. Load takes the values 0.25, 0.60, or 1.00 and is redrawn every 20 cycles. Source probabilities are 0.50, 0.35, and 0.15; target probabilities are 0.15, 0.35, and 0.50. Damage accelerates with both load and accumulated deterioration according to Equation (1). The small multiplicative disturbance represents cycle-level process variation (Eq.1).

$$d_{t+1} = d_t + v(0.5 + 0.9\ell_t)(1 + 0.9d_t^2)\max(0.5, 1 + \varepsilon_t) \quad (1)$$

Here d is dimensionless damage, v is the asset-specific wear coefficient, ℓ is load, and ε is a zero-mean Gaussian variable with standard deviation 0.04. Simulation stops when the next damage value reaches one. Six observation channels combine linear, quadratic, square-root, exponential, and load-interaction functions of damage. Their exact expressions appear in the accompanying reproducibility materials. Independent measurement noise has standard deviation 0.025, and each channel also receives an asset-level Gaussian offset with standard deviation 0.015. These are dimensionless sensor surrogates rather than calibrated vibration, current, or temperature measurements.

Target observations apply channel gains of 1.15, 0.90, 1.05, 1.20, 0.95, and 1.10, followed by offsets of 0.15, −0.10, 0.10, 0.12, −0.08, and 0.05. Missing-reading probabilities increase from 2% in the source domain to 8% in the target domain. Figure 2 shows the load distributions and a prespecified target trajectory. Load changes affect the observations immediately and damage accumulation over time. The target shift therefore changes both the signal-to-health relationship and the expected future time to failure.

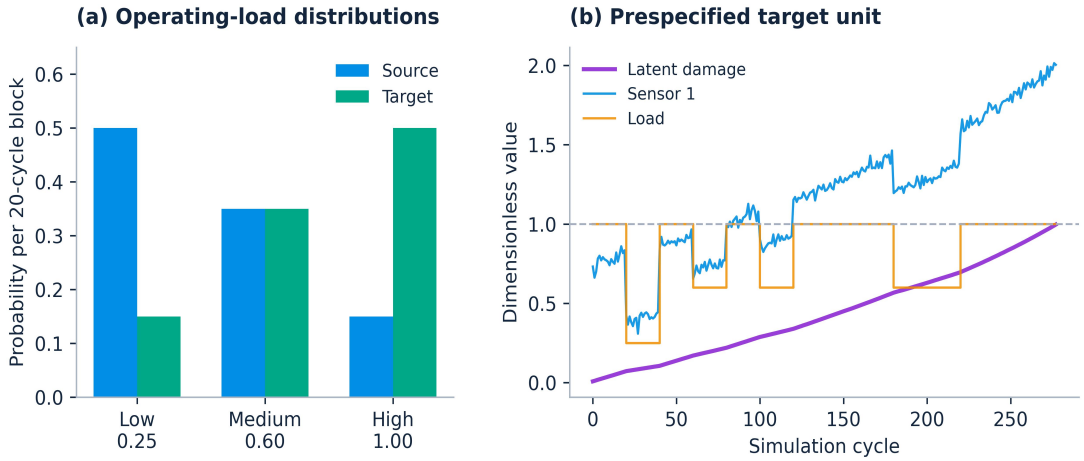


Figure 2. Synthetic operating conditions and degradation. (a) Load probabilities for source and target fleets. (b) Prespecified target test asset zero in repetition zero; the sensor channel combines load and degradation. All plotted observations are simulated.

The paired stress condition removes an additional independent 20% of readings and adds offsets of 0.20 and −0.15 to the first two channels. Combined with the original target missingness, the expected missing fraction is 26.4%, not 28%. Calibration remains based on the ordinary target domain. This is an intentional violation of the matched calibration setting. The simulator contains one continuous degradation mechanism, no maintenance resets, no competing failures, and no censoring. These choices favour clean identification of architectural effects while sharply limiting industrial generalization.

Causal Summaries and Context Variables

Missing observations are filled with the most recent available value from the same channel. Before the first valid reading, the fill value is zero. Availability masks remain explicit. For each trailing window of 8, 16, or 32 cycles, the method computes the mean, standard deviation, linear slope, and last value of each of the six channels. Slopes are multiplied by 32 to express change over a common reference span. Mean load, load standard deviation, and six channel-availability fractions complete a 32-dimensional feature vector at each scale.

The gate receives four context variables from the 32-cycle window: mean load, load standard deviation, final load, and the overall missing fraction. It does not receive asset identity, calendar age, latent damage, or future operating information. Feature and context normalization uses source training means and standard deviations only, with a standard-deviation floor of 0.05. The same transformation is retained during target updating and testing. This keeps preprocessing identical across configurations and avoids allowing target statistics to introduce an unreported form of adaptation.

Temporal summaries make the experiment interpretable but discard waveform detail. A mean and slope cannot preserve every transient, frequency component, or cyclic interaction. That tradeoff is intentional: the study tests whether conditional weighting and restricted updating help a small neural representation. It does not test end-to-end feature learning from high-frequency signals. A production implementation would also need timestamp synchronization and a sampling interval tied to the component's dynamics; a simulation cycle has no claimed equivalence to one minute, one revolution, or one manufacturing operation.

Neural Experts and Conditional Weighting

Each scale feeds an independent neural expert with 32 inputs, one 24-unit rectified-linear hidden layer, and one scalar output. A softplus activation keeps each expert's life estimate positive. The gate uses a four-input, 12-unit hyperbolic-tangent hidden layer and a three-output softmax. Equation (2) combines the expert estimates. Network outputs are expressed in hundreds of simulation cycles during optimization and multiplied by 100 when reporting performance. The architecture has 2,550 active parameters, including 99 in

the gate .

$$\hat{r}_t = 100 \sum_{k=1}^{3} w_k(c_t) f_k(z_{k,t}) \quad (2)$$

$$w_k(c_t) = \frac{\exp g_k(c_t)}{\sum_{j=1}^{3} \exp g_j(c_t)} \quad (3)$$

In Equation (2 and 3), z_k is the normalized summary at temporal scale k, c is the normalized context, f_k is the corresponding expert, and g_k is its gate logit. Positive normalized weights create a convex combination of positive expert estimates. The uniform configuration fixes all three weights at one third and has 2,451 active parameters. It retains the same experts, temporal summaries, input information, optimizer budget, and data partitions. Thus, the gating comparison changes only conditional weighting and its associated 99 parameters.

The network performs soft weighting rather than sparse routing. Every expert is evaluated for every prediction, so the gate does not reduce inference work by skipping branches. Nor does the model grow new experts as conditions change. Its adaptivity consists of context-dependent weights at inference and a separately controlled parameter update. Gate weights can be inspected to diagnose sensitivity to operating context, but the design provides no basis for interpreting them as probabilities of physical failure modes.

Training and Restricted Target Updating

Source training minimizes mean squared error on normalized RUL, with an L2 penalty coefficient of 0.00001 applied to active parameters. Adam uses learning rate 0.003, moment coefficients 0.9 and 0.999, numerical constant 10^{-8}, and batches of up to 128 windows [26]. Training runs for 100 epochs, and the checkpoint with the lowest source-validation mean squared error is retained. Both configurations begin with matching expert initialization within each repetition. Hyperparameters are fixed across repetitions; no test-driven search is performed.

The restricted update freezes the three expert hidden layers and modifies only their output weights and biases, together with the gate when present. This gives 75 updateable parameters in the uniform model and 174 in the gated model, respectively 3.1% and 6.8% of their active totals. Each target minibatch is supplemented with approximately one source window for every three target windows. The objective includes squared deviation from the source parameters with coefficient 0.001. This anchoring and source rehearsal discourage unnecessary departure from the source solution, although neither guarantees retention.

Updating uses learning rate 0.002 for 100 epochs. The ten target-validation assets select the checkpoint independently of the calibration assets. A source-trained model is copied before updating, so frozen and updated results can be paired. The four configurations are uniform frozen, gated frozen, uniform updated, and gated updated. All form ablations of the same method. The update is supervised and retrospective: full failure histories are assumed available before a target model is approved. No online weight changes, autonomous pseudo-labelling, or automatic maintenance actions are implemented.

Calibration and Performance Measures

After checkpoint selection, each configuration is calibrated separately using its absolute endpoint errors on the relevant domain's 120 calibration assets. For nominal 90% coverage, the half-width is the 109th smallest residual, since the finite-sample index is the ceiling of 121 multiplied by 0.90. Equation (4) defines the prediction interval. Truncating its lower endpoint at zero respects the nonnegative target without excluding any admissible positive RUL value. The stressed target observations reuse the ordinary target half-width without recalibration.

$$C_t = [\max(0, \hat{r}_t - q_{0.9}), \hat{r}_t + q_{0.9}], q_{0.9} = a_{(109)} \quad (4)$$

The one-endpoint-per-asset design avoids treating strongly overlapping windows as independent

calibration cases. Under the simulator's matched asset-generation and endpoint-sampling rules, calibration and test cases are exchangeable conditional on the trained predictor. The resulting statement concerns marginal coverage for that endpoint population. It does not promise simultaneous coverage of an entire trajectory, conditional coverage for low-life assets, or validity after an additional domain shift. Even frozen predictors receive labelled target calibration. Their interval results must therefore not be described as label-free transfer.

Evaluation reports root mean squared error, mean absolute error, signed bias, empirical interval coverage, and mean interval width. Positive bias means overestimating available life. Two additional diagnostics examine endpoints with true RUL at most 60 cycles and the fraction of predictions exceeding true RUL by more than 20 cycles. These thresholds are fixed descriptive choices, not validated maintenance limits. All headline values average the five repetition-level statistics; reported standard deviations describe variation across repetitions. Paired differences are used when comparing configurations, and individual windows are not treated as independent experimental replications.

Reproducibility and Experimental Boundaries

The implementation uses NumPy and explicit neural-network derivatives. A finite-difference check covers representative entries in every parameter tensor before training. The reproducibility package contains the simulator, architecture, optimization procedure, random seeds, endpoint predictions, repetition-level metrics, and plotting code. Figures are generated directly from those saved outputs. The illustrative trajectory is fixed as target test asset zero in repetition zero and is not selected for visual agreement. Source and target partitions use distinct random streams.

This protocol is sufficient to reproduce the stated synthetic comparisons. It does not establish robustness to unknown degradation mechanisms, demonstrate calibrated uncertainty in a real plant, or measure economic benefit. No public dataset is used for the reported results, and no runtime measurement is interpreted as an edge-device benchmark. The restricted architecture and simple simulator are methodological choices. Their value is that the numerical claims can be checked without confusing a simulation result with an industrial observation.

The experimental unit for uncertainty about performance is the independently generated repetition. Within a repetition, the four configurations share data and expert initialization, making their differences paired. Across repetitions, both asset histories and network initialization change. This combines sampling and optimization variability rather than separating them. A dedicated sensitivity study would require additional repetitions with fixed fleets and varied initialization, followed by independently varied fleets. The current standard deviations should therefore be read as descriptive variation under the complete protocol, not as confidence limits for an industrial population.

RESULTS

Prediction Accuracy and the Contribution of Updating

Figure 3 compares source and target errors, while Table 2 reports the target-domain metrics. The frozen uniform model achieves a mean target RMSE of 29.93 cycles. The frozen gated model increases that error to 31.94 cycles, despite having a slightly lower source RMSE, 28.16 versus 28.41 cycles. The gate therefore offers no reliable transfer advantage before updating. Its target error is lower than the uniform model in only two of the five repetitions. The source result alone would have concealed this weakness.

Updating reduces target RMSE to 25.29 cycles for uniform weighting and 24.23 cycles for gating. Relative to their own frozen counterparts, these are reductions of 15.5% and 24.2%, respectively. The gated model improves after updating in all five repetitions. Much of the apparent benefit of the full adaptive configuration thus comes from receiving labelled target histories and performing additional optimization. The appropriate architectural comparison is between the two updated configurations, not between the updated gate and a frozen baseline denied that information.

For that comparison, the mean RMSE difference is 1.07 cycles in favour of gating, equivalent to 4.2% of the updated uniform error. Gating wins in four repetitions, with paired differences ranging from a 3.04-cycle advantage to a 0.55-cycle disadvantage. An illustrative paired t interval for the mean difference, gated minus uniform, spans approximately −2.82 to 0.69 cycles. Five repetitions provide limited precision, and this interval assumes approximately normal repetition-level differences. The evidence supports a modest average improvement within the simulator, not a dependable general advantage.

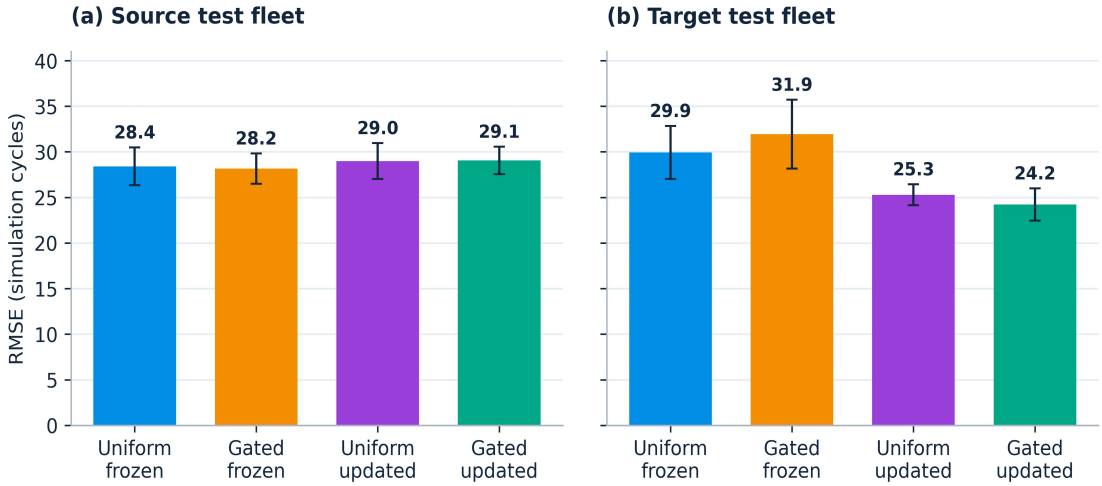


Figure 3. Prediction error across five synthetic repetitions. Bars show mean RMSE; whiskers show one sample standard deviation across repetitions. Source and target test sets each contain 160 independent asset endpoints per repetition.

Table 2. Target Prediction and Interval Performance across Five Repetitions

Configuration	RMSE cycles	MAE cycles	Bias cycles	Coverage %	Width cycles
Uniform frozen	29.93 ± 2.90	21.58	-3.49	90.3	93.69
Gated frozen	31.94 ± 3.78	24.48	1.05	87.8	96.67
Uniform updated	25.29 ± 1.14	17.98	-2.01	89.4	81.82
Gated updated	24.23 ± 1.76	17.86	-1.89	90.9	83.09

Note: RMSE is mean ± sample standard deviation across repetitions. Other values are repetition means. Intervals have nominal 90% marginal coverage. Negative bias indicates underestimation.

Source Retention and Errors Near Failure

Restricted updating does not entirely preserve source performance. Uniform source RMSE rises from 28.41 to 28.99 cycles; gated source RMSE rises from 28.16 to 29.06 cycles. These increases are small in absolute terms but show that freezing hidden layers, anchoring parameters, and replaying source examples do not eliminate interference. Because these safeguards are always used together, the experiment cannot identify their individual effects or establish that they outperform unconstrained fine-tuning. It only measures retention under the specified update procedure.

Near failure, the ranking differs from the overall target ranking. At endpoints with at most 60 cycles remaining, the updated uniform model has mean RMSE 8.75 cycles, compared with 9.89 cycles for the updated gated model. The latter nevertheless improves substantially over the frozen gate's 17.53 cycles. There are 197 such target endpoints across the five repetitions. Their lower errors partly reflect the

simulator's increasingly informative damage signals near failure. This subgroup result cautions against selecting a maintenance model solely by aggregate RMSE.

Calibration and the Illustrative Life Trajectory

The updated gated model achieves mean target interval coverage of 90.9%, with mean width 83.09 cycles. The updated uniform model produces 89.4% coverage and width 81.82 cycles. Both are close to the nominal marginal level in this matched setting, although finite calibration and test samples produce repetition-to-repetition variation. Lower prediction error does not automatically yield the narrowest empirical interval: calibration depends on the residual tail, while RMSE summarizes squared errors across the whole test population.

Figure 4 displays the prespecified target trajectory and its learned expert weights. The life estimate changes as operating context changes, and the weights reveal how the architecture redistributes computational influence. The interval is intentionally broad. It uses a domain-level residual half-width rather than a locally learned uncertainty scale. Its plotted sequence is a diagnostic illustration; the experiment has not calibrated simultaneous trajectory coverage. An entire curve lying inside its band would not establish such a guarantee, just as a local miss would not alone refute the endpoint-level calibration claim.

All late-life target endpoints are covered by both updated models, averaging 100% coverage within this subgroup. That apparent success accompanies conservative interval widths relative to short remaining lives. It should not be advertised as unusually precise late-life uncertainty. A model can cover nearly every short-lived asset by issuing intervals too broad to distinguish replacement today from replacement several weeks later. The study reports coverage and width together because either quantity alone invites an incomplete interpretation.

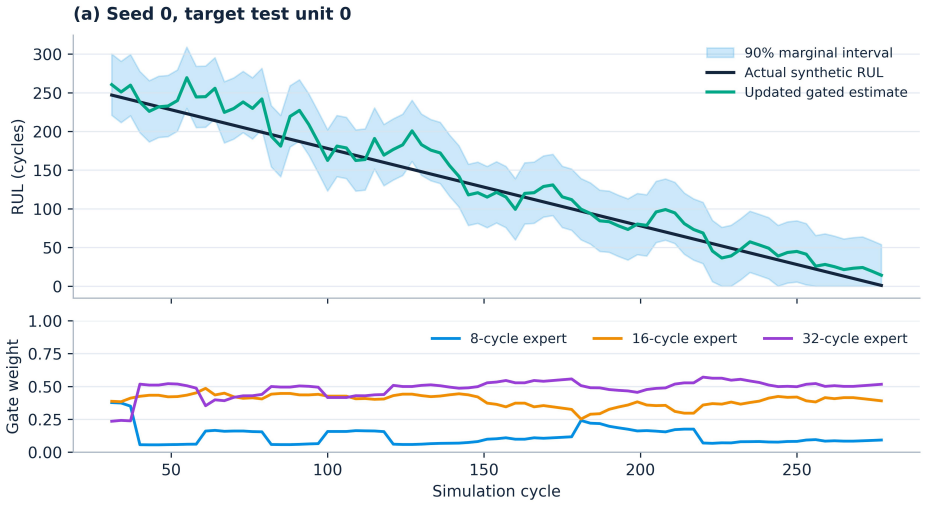


Figure 4. Prespecified synthetic target trajectory for the updated gated model. The band uses the target-calibrated 90% marginal endpoint interval and does not claim simultaneous trajectory coverage. Lower-panel weights indicate computational influence, not physical failure-mode probabilities.

Additional Sensor Faults Expose a Boundary

Table 3 and Figure 5 show the paired sensor-stress results. The updated gated model's RMSE rises from 24.23 to 42.23 cycles, and coverage falls from 90.9% to 73.6%. Its mean interval width barely changes, from 83.09 to 83.17 cycles. This is expected: the calibration half-width is unchanged, and width differences arise only from zero truncation. A fixed interval cannot diagnose that the error-generating conditions have changed. The gate's access to missingness information is insufficient protection against additional missingness combined with systematic offsets.

Table 3. Late-Life Target Accuracy and Paired Sensor-Stress Performance.

Configuration	Late target RMSE cycles	Stress RMSE cycles	Stress coverage %	Stress bias cycles
Uniform frozen	11.06	52.05 ± 6.94	68.0	-21.88
Gated frozen	17.53	67.19 ± 13.87	56.7	-18.85
Uniform updated	8.75	40.68 ± 7.44	73.6	-15.32
Gated updated	9.89	42.23 ± 7.44	73.6	-17.66

Note: Late life means true RUL ≤ 60 cycles (197 target endpoints across five repetitions). Stress adds 20% independent reading removal and offsets to two channels. Target calibration is retained. Values are repetition means; ± indicates one sample standard deviation.

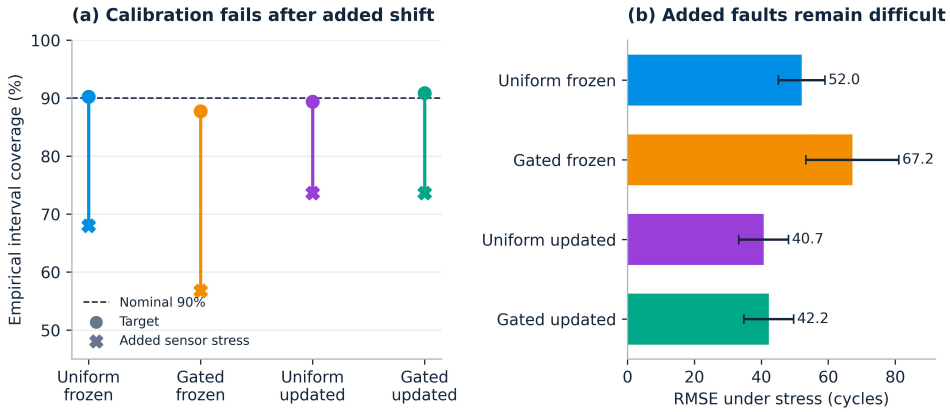


Figure 5. Paired sensor-stress evaluation. (a) Mean coverage before and after added missingness and offsets, without recalibration. (b) Mean stressed RMSE with one standard deviation across five repetitions. The target failure times and test endpoints remain fixed.

The updated uniform model performs better under stress, with RMSE 40.68 cycles and the same rounded mean coverage of 73.6%. The frozen gated model is most vulnerable, reaching 67.19 cycles RMSE and 56.7% coverage. Updated configurations retain useful advantages over their frozen versions, but neither maintains the intended uncertainty performance. Their stressed biases are negative, approximately − 15.32 and − 17.66 cycles for uniform and gated weighting, respectively. These averages indicate premature life estimates overall; they do not rule out dangerous overestimation for individual assets.

DISCUSSION

Adaptation Helps When its Information is Relevant

The clearest result concerns supervised updating, not architectural novelty. Both configurations benefit from labelled target histories. The gate's small additional advantage under the ordinary target shift is plausible because operating load changes the relationship between short-term observations and future degradation. Yet the same gate performs worse without updating and loses its advantage under sensor stress. Conditional computation is therefore an opportunity to learn a useful dependence, accompanied by an opportunity to learn the wrong one. The experiment supports neither automatic rejection nor automatic preference for gating.

The uncomfortable comparison is often the informative one. A relatively simple uniform mixture remains competitive after receiving the same target data and optimization budget. That comparison changes the practical interpretation of a 24.2% improvement against the frozen gate. It is an update benefit within an architecture, not evidence that adaptive weighting is 24.2% better than a fairly trained alternative. For engineering decisions, the incremental benefit must justify the extra monitoring, validation, and configuration burden. A small numerical gain may still matter, but its value depends on the decision it changes.

The adaptation literature provides richer alternatives to the restricted update studied here, including operation-specific alignment and feature disentanglement [10], [12]. Their additional structure may be warranted when target labels are scarce or shifts are more complex. This study cannot rank those alternatives. It instead provides a transparent lower-complexity comparison that a broader benchmark could include. Any proposed replacement should be evaluated with the same target information budget and explicit treatment of calibration labels.

Calibration is Conditional on the Evaluation Contract

The stress result is not a failure of the split conformal theorem. The test distribution has been changed after calibration. What fails is the operational assumption that past residuals remain representative. This distinction is essential when uncertainty statements move from a paper into a dashboard. A label such as 90% prediction interval can sound like an enduring property of a model. Here, it is a property of a predictor, a calibration population, and an endpoint-selection rule considered together. Altering one element can invalidate the interpretation.

More sophisticated conformal procedures can address some distribution changes, but their assumptions and feedback requirements remain relevant [22], [23]. RUL presents a particular delay: the true outcome may become observable only after failure or a sufficiently informative inspection. A stream of sensor values is not a stream of verified prediction errors. An operational system therefore needs a policy for uncertainty statements during periods without trustworthy outcome feedback. The appropriate response may be inspection or temporary reliance on established maintenance limits rather than immediate statistical recalibration.

The constant-width interval also exposes a mismatch between marginal validity and useful decisions. Early-life errors and late-life errors contribute to a common residual quantile, making intervals conservative near failure. Conformalized quantile approaches offer one way to adapt interval scale [19], [20], but they would introduce another modelling decision and require their own evaluation. Conditional or subgroup performance should be reported explicitly rather than inferred from an attractive fleet-wide coverage percentage. A useful interval needs both credible coverage and enough resolution to affect an action.

From Life Estimates to Maintenance Decisions

A maintenance decision would combine RUL information with lead times, spare-part availability, shutdown opportunities, and the cost of failure. None of those quantities is estimated here. The 20-cycle overprediction diagnostic and 60-cycle late-life subgroup provide transparent views of error behaviour, but they are not optimized intervention thresholds. An asymmetric loss could prioritize avoiding late intervention, yet it could also create excessive replacement if its cost assumptions were wrong. Choosing the objective requires maintenance knowledge that a generic sensor dataset does not supply.

The differing late-life ranking is particularly relevant. The updated gated model has lower overall RMSE, while the updated uniform model predicts the selected near-failure subgroup more accurately. A production team concerned mainly with short-horizon scheduling could reasonably prefer the latter. Another team planning inventory months ahead might weight early-life errors more heavily. Model selection should specify that decision horizon before comparing algorithms. Otherwise, the preferred metric tends to change after the results are known, and an apparently rigorous benchmark becomes difficult to interpret.

For a deployment study, the proposed predictor would initially run in shadow mode. Each estimate would be logged with sensor availability, operating context, interval width, and model version. Engineers would compare those records with inspections and maintenance outcomes before allowing recommendations to influence scheduling. A candidate target update would undergo source-retention and target-validation

checks, with a documented rollback option. These are proposed operating arrangements, not implemented capabilities or validated safety controls. The present experiment provides prediction evidence only.

Data Availability and Computational Practicality

The parameter count is encouraging for implementation: updating 174 parameters is a modest optimization problem. It does not establish low deployment cost. Sensor synchronization, feature calculation, model-version management, and outcome reconciliation can outweigh the arithmetic of a small network. Because all three experts execute at inference, this architecture should not be described as computationally sparse. Hardware latency, memory use, energy demand, and end-to-end reliability remain unmeasured. Parameter counts are an architectural property, not a substitute for system benchmarking.

The information budget is a stronger constraint. Although only 30 target histories train the update, validation and calibration require another 130 labelled target assets. Many factories will not have that number of comparable run-to-failure histories, particularly for high-value equipment maintained before failure. Active selection can improve the use of available labels [24], but it cannot turn a censored maintenance record into an observed failure time without additional assumptions. A practical extension would need to handle censoring, changes in failure definitions, and selection introduced by existing maintenance policies.

5.5. Limits and the Next Evidentiary Step

The simulator intentionally resembles the method's preferred conditions: smooth degradation, informative temporal summaries, known operating context, and enough complete histories for updating and calibration. Different degradation mechanisms could favour different representations. Abrupt faults, intermittent symptoms, maintenance resets, and correlated sensor failures are absent. The stress test changes measurement quality but preserves the latent wear law, so it does not establish resilience to mechanism change. Five repetitions quantify some numerical variation, yet they cannot represent the diversity of industrial assets or all plausible simulator settings.

The study also leaves the relative importance of its three temporal scales unresolved. Equal-weight and learned-weight mixtures test the gate, but neither reveals whether a single well-tuned scale could match their performance. Likewise, the paired sensor stress combines missingness with offsets, so its degradation cannot be assigned uniquely to either disturbance. Those are useful next ablations rather than explanations established by the present results. Keeping these unanswered questions explicit prevents the architecture diagram from implying that every component has earned independent experimental support.

The next evidentiary step is a preregistered evaluation on measured machinery histories and established public datasets, with assets separated before preprocessing. Comparators should include a strong fixed predictor, a full-update version, and relevant contemporary domain-adaptation methods. Label budgets, checkpoint selection, and calibration resources must match. Failure mechanisms and operating regimes should be reported separately, with decision-focused outcomes alongside prediction error. Such a study could determine whether the modest gating advantage survives outside this controlled setting. Until then, the most defensible conclusion concerns disciplined adaptation and honest validation, not readiness for autonomous maintenance.

CONCLUSION

This paper developed and tested a compact adaptive neural method for RUL estimation using causal summaries, three temporal experts, context-dependent weighting, restricted supervised updating, and independent residual calibration. The five-repetition synthetic experiment separates the contribution of gating from the value of labelled target data. Updating improves prediction in both architectural configurations. The updated gated model reaches 24.23 cycles mean target RMSE, but its advantage over the equally updated uniform configuration is only 1.07 cycles and is uncertain across the small set of repetitions.

The reliability results impose a clearer limit. Nominal 90% intervals achieve 90.9% mean coverage under the matched target setting and only 73.6% after additional sensor faults. The updated uniform model also has lower stressed error and lower late-life target error. These reversals show why overall accuracy, subgroup performance, and interval calibration should be examined together. Adaptive weighting can help a

model use operating context, but it cannot establish that the context remains informative or that its calibration population remains representative.

The architecture is computationally modest, with 2,550 parameters and 174 updated parameters, yet its label requirements remain substantial. The evidence supports controlled target updating as a useful option within this simulator and a transparent framework for subsequent testing. Industrial usefulness will require measured degradation histories, realistic censoring, fair external comparisons, and explicit maintenance decisions. A credible prognostic model should make those requirements visible. Confidence is earned by knowing where an estimate remains dependable and by recognizing when the evidence no longer supports it.

REFERENCES

[1] O. Serradilla, E. Zugasti, J. Rodriguez, and U. Zurutuza, “Deep learning models for predictive maintenance: A survey, comparison, challenges and prospects,” *Applied Intelligence*, vol. 52, pp. 10934–10964, 2022. doi: 10.1007/s10489-021-03004-y.

[2] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, “Machinery health prognostics: A systematic review from data acquisition to RUL prediction,” *Mechanical Systems and Signal Processing*, vol. 104, pp. 799–834, 2018. doi: 10.1016/j.ymssp.2017.11.016.

[3] M. Arias Chao, C. Kulkarni, K. Goebel, and O. Fink, “Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics,” *Data,* vol. 6, no. 1, Art. no. 5, 2021. doi: 10.3390/data6010005.

[4] Y. Hou et al., “Evidential domain adaptation for remaining useful life prediction with incomplete degradation,” *IEEE Transactions on Instrumentation and Measurement*, vol. 74, pp. 1–11, 2025. doi: 10.1109/TIM.2025.3551977.

[5] L. Zhou and H. Wang, “An adaptive multi-scale feature fusion and adaptive mixture-of-experts multi-task model for industrial equipment health status assessment and remaining useful life prediction,” *Reliability Engineering & System Safety*, vol. 248, Art. no. 110190, 2024. doi: 10.1016/j.ress.2024.110190.

[6] X. Li, Q. Ding, and J.-Q. Sun, “Remaining useful life estimation in prognostics using deep convolution neural networks,” *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018. doi: 10.1016/j.ress.2017.11.021.

[7] Z. Xu, Y. Zhang, and Q. Miao, “An attention-based multi-scale temporal convolutional network for remaining useful life prediction,” *Reliability Engineering & System Safety*, vol. 250, Art. no. 110288, 2024. doi: 10.1016/j.ress.2024.110288.

[8] H. Li, Z. Zhang, T. Li, and X. Si, “A review on physics-informed data-driven remaining useful life prediction: Challenges and opportunities,” *Mechanical Systems and Signal Processing*, vol. 209, Art. no. 111120, 2024. doi: 10.1016/j.ymssp.2024.111120.

[9] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, “Adaptive mixtures of local experts,” *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991. doi: 10.1162/neco.1991.3.1.79.

[10] I. Nejjar, F. Geissmann, M. Zhao, C. Taal, and O. Fink, “Domain adaptation via alignment of operation profile for Remaining Useful Lifetime prediction,” *Reliability Engineering & System Safety*, vol. 242, Art. no. 109718, 2024. doi: 10.1016/j.ress.2023.109718.

[11] Z. Lei, Y. Su, K. Feng, and G. Wen, “Interpretable operational condition attention-informed domain adaptation network for remaining useful life prediction under variable operational conditions,” *Control Engineering Practice*, vol. 153, Art. no. 106080, 2024. doi: 10.1016/j.conengprac.2024.106080.

[12] J. Yan, Z.-S. Ye, S. He, and Z. He, “A feature disentanglement and unsupervised domain adaptation of remaining useful life prediction for sensor-equipped machines,” *Reliability Engineering & System Safety*, vol. 242, Art. no. 109736, 2024. doi: 10.1016/j.ress.2023.109736.

[13] S. Lv, C. Xia, C. Cheng, J. Yan, and X. Wu, “Label adversarial domain adaptation network for predicting remaining useful life based on cross-domain condition,” *Computers & Industrial Engineering*, vol. 197, Art. no. 110542, 2024. doi: 10.1016/j.cie.2024.110542.

[14] R. Mo, H. Zhou, H. Yin, and X. Si, “A survey on few-shot learning for remaining useful life prediction,” *Reliability Engineering & System Safety*, vol. 257, Part B, Art. no. 110850, 2025. doi: 10.1016/j.ress.2025.110850.

[15] E. Hüllermeier and W. Waegeman, “Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods,” *Machine Learning*, vol. 110, pp. 457–506, 2021. doi: 10.1007/s10994-021-05946-3.

[16] M. Benker, L. Furtner, T. Semm, and M. F. Zaeh, “Utilizing uncertainty information in remaining useful life estimation via Bayesian neural networks and Hamiltonian Monte Carlo,” *Journal of Manufacturing Systems*, vol. 61, pp. 799–807, 2021. doi: 10.1016/j.jmsy.2020.11.005.

[17] J. Yang, Y. Peng, J. Xie, and P. Wang, “Remaining useful life prediction method for bearings based on LSTM with uncertainty quantification,” *Sensors*, vol. 22, no. 12, Art. no. 4549, 2022. doi: 10.3390/s22124549.

[18] A. Javanmardi and E. Hüllermeier, “Conformal prediction intervals for remaining useful lifetime estimation,” *International Journal of Prognostics and Health Management*, vol. 14, no. 2, 2023. doi: 10.36001/ijphm.2023.v14i2.3417.

[19] Y. Romano, E. Patterson, and E. J. Candès, “Conformalized quantile regression,” in A*dvances in Neural Information Processing Systems*, vol. 32, 2019. [Online]. Available: https://papers.neurips.cc/paper_files/paper/2019/hash/5103c3584b063c431bd1268e9b5e76fb-Abstract.html

[20] S. Piao, R. Huang, and F. Tsung, “CRULP: Reliable RUL estimation inspired by conformal prediction,” *IEEE Transactions on Instrumentation and Measurement*, vol. 74, Art. no. 3505411, 2025. doi: 10.1109/TIM.2024.3522678.

[21] A. N. Angelopoulos and S. Bates, “Conformal prediction: A gentle introduction,” *Foundations and Trends in Machine Learning*, vol. 16, no. 4, pp. 494–591, 2023. doi: 10.1561/2200000101.

[22] R. F. Barber, E. J. Candès, A. Ramdas, and R. J. Tibshirani, “Conformal prediction beyond exchangeability,” *The Annals of Statistics*, vol. 51, no. 2, pp. 816–845, 2023. doi: 10.1214/23-AOS2276.

[23] I. Gibbs and E. J. Candès, “Adaptive conformal inference under distribution shift,” in *Advances in Neural Information Processing Systems*, vol. 34, 2021. [Online]. Available: https://papers.neurips.cc/paper_files/paper/2021/hash/0d441de75945e5acbc865406fc9a2559-Abstract.html

[24] Z. Kharazian, T. Lindgren, S. Magnusson, and H. Boström, “CoPAL: Conformal prediction in active learning an algorithm for enhancing remaining useful life estimation in predictive maintenance,” in *Proceedings of the Thirteenth Symposium on Conformal and Probabilistic Prediction with Applications, Proceedings of Machine Learning Research*, vol. 230, pp. 195–217, 2024. [Online]. Available: https://proceedings.mlr.press/v230/kharazian24a.html

[25] W. Wang, Z. Wang, Z. Cai, C. Hu, and S. Si, “Robust uncertainty quantification for online remaining useful life prediction with randomly missing and partially faulty sensor data,” *Reliability Engineering & System Safety*, vol. 262, Art. no. 111177, 2025. doi: 10.1016/j.ress.2025.111177.

[26] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2015. [Online]. Available: https://arxiv.org/abs/1412.6980