aerospace

MDPI

Article

Validation-Selected Constrained Incremental Learning for Simulation-Based UAV Subsystem Fault Prediction

Guang Rong [1,2], Di Wu [3], Chao Tang [1], Ye Huang [4], Xiangyu Meng [1,*], Jian Wang [1,2] and Jianyuan Wang [1,2]

[1] College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China; rock_mizuno@163.net (G.R.); 18063009650@163.com (C.T.); bx2315004@nuaa.edu.cn (J.W.); wangjyuan@nuaa.edu.cn (J.W.)

[2] National Key Laboratory of Aerospace Mechanism, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

[3] China Rong Tong Academy of Sciences Group Corporation Limited, Beijing 100011, China; wudi_uestc@163.com

[4] Information Center, National Defense Mobilization Department, Central Military Commission, Beijing 100034, China; breezeleaf@126.com

* Correspondence: mengxiangyu@nuaa.edu.cn

Abstract

Fixed offline prognostic models can lose accuracy when sensor-data distribution changes after model development. This paper presents a validation-selected incremental-learning framework for time-before-failure (TBF) regression in three simulated unmanned-aerial-vehicle (UAV) subsystems: avionics, power, and airframe structure. According to the dataset provider, the subsystem simulation records used in this paper were generated with a customized quadrotor model on the RflySim framework, and the original source arrays used in the experiments are publicly available through the research repository specified in the Data Availability Statement. A leakage-controlled protocol partitions 200 parameterized degradation realizations per subsystem before window use, fits normalization statistics on source-training realizations only, and creates a controlled target domain through standardized sensor bias, gain, noise, and contiguous dropout. CNN–LSTM, TCN–CNN, and Transformer–CNN candidates are compared over three random seeds; CNN–LSTM is selected for all three subsystems by the source-validation RMSE. The incremental benchmark compares no update, fine tuning, elastic weight consolidation (EWC), replay, learning without forgetting (LwF), and an L2 parameter constraint. Each trainable method receives four predeclared candidates, the same three seeds and five-epoch budget, and selection by the same old-plus-new validation score before held-out testing. The resulting stability–plasticity balance is subsystem and storage-condition dependent: replay gives the strongest balance for power and airframe when source storage is permitted, whereas tuned fine tuning gives the lowest forgetting for avionics among the replay-free updates. L2 remains a compact replay-free update but is not universally superior to the other tuned methods. Near-failure errors and threshold warning rates provide an additional maintenance-oriented acceptance check. The evidence establishes controlled simulation feasibility; real-flight, bench-test, and hardware-in-the-loop validation remain outside the present scope.

Keywords: unmanned aerial vehicle; fault prediction; incremental learning; catastrophic forgetting; predictive maintenance; remaining useful life; time-before-failure; domain shift

Check for updates

Academic Editors: Bosko Rasuo and Gang Sun

Received: 8 July 2026
Revised: 22 August 2026
Accepted: 27 August 2026
Published: 28 August 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Unmanned aerial vehicles (UAVs) combine sensing, computation, propulsion, control, and load-bearing structures in a tightly coupled platform. A fault that develops in one subsystem can interrupt a mission, trigger unplanned maintenance, or propagate into a safety-critical condition. Prognostic health management therefore requires more than recognizing an existing fault: it requires an estimate of the remaining lead time before a defined failure event [1–3]. In this paper, that lead time is represented by time before failure (TBF), which is a continuous remaining-useful-life-type target.

Deep sequence models provide effective representations for multichannel degradation trajectories. Recurrent networks retain long temporal context, convolutional networks extract local changes, temporal convolutional networks enlarge the receptive field through dilation, and Transformer encoders relate distant time positions [4–7]. Their performance, however, is normally established under a fixed training distribution. Sensor calibration changes, component replacement, environmental exposure, and aging can alter the observed feature distribution after the original model has been trained. Updating on the new distribution introduces the stability–plasticity problem: the model must acquire the new relation without discarding behavior that remains useful on the source distribution [8,9].

Continual-learning strategies approach this problem through replay, regularization, distillation, gradient constraints, or architecture expansion [10–13]. Replay directly mixes stored source samples with new samples, whereas parameter regularization can update without retaining historical windows. For UAV prognostics, that distinction is operationally relevant because historical sensor windows may be large, licensed, or governed by fleet-data policies. A direct L2 constraint is consequently evaluated here as a compact replay-free update—not as a universal replacement for continual-learning methods.

Recent prognostic and diagnostic research further motivates explicit uncertainty, transfer, and limited-label evaluation. A Wiener-prior-guided hierarchical network integrates degradation priors with uncertainty quantification for aero-engine RUL prediction [14]. Spatial–channel graph interaction has been used for unsupervised transfer across rotating-machinery domains [15]. Dynamic prototype contrastive learning addresses semi-supervised diagnosis when labeled target samples are scarce [16]. These studies show that strong predictive performance must be interpreted together with the information available during adaptation. This paper focuses on a complementary setting: batch domain-incremental TBF regression with an unchanged target definition, explicit old/new validation, and no source-window storage for the proposed update.

This paper makes four contributions:

1. It defines a reproducible realization-level protocol for three simulated UAV TBF subsystems, including window geometry, split counts, source-only normalization, and a quantitative controlled sensor-distribution shift.
2. It separates architecture selection, method-specific hyperparameter selection, and held-out testing. Architectures are chosen from source validation, and every trainable update method receives four candidates selected from old/new validation only.
3. It measures stability and plasticity directly through held-out old-domain forgetting and new-domain adaptation gain, and it compares the L2 update with no update, fine tuning, EWC, replay, and LwF over three random seeds under a budget-matched validation-only selection protocol.
4. It extends aggregate regression evaluation with near-failure error distributions and 1, 2, and 3 h missed- and false-warning rates, preventing a high global R^2 from serving as the sole maintenance criterion.

https://doi.org/10.3390/aerospace13090777

All experiments are offline. The controlled perturbation defines a sensor-feature distribution shift for comparative simulation; it is not presented as a surrogate for flight, bench, or hardware-in-the-loop (HIL) evidence.

2. Related Work

2.1. TBF and RUL Prediction

Condition-based maintenance estimates system health from observed signals and schedules intervention according to predicted degradation rather than a fixed service interval [1,17]. RUL methods range from stochastic degradation models to deep temporal networks [2,3]. UAV prognostics add heterogeneous subsystem dynamics and multichannel coupling. Electrical and avionics measurements can contain abrupt local changes, while structural responses can accumulate over longer intervals. TBF regression retains the continuous maintenance lead time that a health-state classifier does not provide.

An aggregate metric alone does not determine whether a prognostic model is suitable for maintenance. MSE and RMSE quantify absolute error, while R^2 normalizes error by target variance. Near-failure behavior is especially important because a modest error far from failure and the same error close to a maintenance threshold have different consequences. This motivates the combined use of global regression metrics, near-failure errors, and explicit threshold decisions.

2.2. Domain-Incremental Learning

In domain-incremental learning, the input distribution changes while the target space remains fixed [9]. Fine tuning provides maximal plasticity but does not explicitly preserve source behavior. EWC weights parameter changes by an estimate of source-task importance [11]. Replay preserves source examples and is often a strong reference when storage is permitted [10,18]. LwF uses the pre-update model as a teacher on the update samples [12]. Gradient-projection methods and expandable architectures provide further alternatives [13,19].

For continuous TBF regression, a direct L2 parameter constraint provides a transparent baseline for conservative adaptation. Its value depends on selecting the constraint strength from validation data and measuring performance on both domains after the update. The revised benchmark is designed around those two requirements.

3. Materials and Methods

3.1. Data Source and Simulation Basis

The UAV subsystem records used in this paper are simulation data released as a research dataset. The original source arrays used for the reported experiments are publicly available through Zenodo: https://doi.org/10.5281/zenodo.22058477. According to the dataset provider, the records were generated using a customized quadrotor model on the RflySim framework developed by the Reliable Flight Control Group at Beihang University [20,21]. RflySim provides a modular environment for multicopter dynamics, flight-control integration, fault/degradation scenario configuration, automated simulation, and data logging. Preparing the present records required configuring the customized vehicle and subsystem models, integrating their closed-loop flight-control interfaces, defining degradation/failure cases and monitored signals, synchronizing multirate outputs, executing long-duration simulation batches through the terminal failure event, and exporting the resulting subsystem records for learning. These records are a project-specific derivative rather than the public RflySimSafe ST001–ST029 files.

The DOI-based research repository provides the original source arrays used in this paper and permits independent inspection of the subsequent realization reconstruction

and data-partitioning procedure. It does not include the complete executable customized RflySim project or a platform-version manifest. Accordingly, public access to the arrays improves dataset-level transparency and reproducibility but is not presented as independent validation of the customized simulator configuration itself. The claims and validation in this paper therefore remain restricted to the shared simulation records and the controlled offline benchmark described below.

3.2. Processed Data Representation and TBF Target

The released dataset contains avionics, power, and airframe-structure TBF regression data. Each subsystem contains 200 parameterized degradation realizations and 10 numeric feature channels. A realization identifier is recovered from the fixed array organization together with the positive reset in the descending TBF sequence. Each record terminates at its designated simulation failure event; the label for a window is the elapsed time from that window to the terminal event, which is expressed in hours. Each sample is a tensor $x \in \mathbb{R}^{300\times 10}$ paired with this continuous TBF label y. The supplied learning package does not include a verified semantic dictionary for the selected ten channels; this paper therefore reports the auditable tensor-level specification and does not attach unverified physical names to individual inputs.

Windows have a length of 300 and step of 100, giving 66.7% overlap within a realization. Avionics and airframe realizations contain 247 windows each; power realizations contain 497. The realization, rather than an overlapping window, is the partition unit. This construction does not interpret the 200 realizations as 200 independently executed flights. Table 1 lists the resulting counts.

Table 1. Realization-level split and window counts. The six partitions are disjoint in realization identity.

Subsystem	Source Train	Source Val.	Old Test	Target Update	Target Val.	Target Test
Realizations	120	20	20	16	4	20
Avionics windows	29,640	4940	4940	3952	988	4940
Power windows	59,640	9940	9940	7952	1988	9940
Airframe windows	29,640	4940	4940	3952	988	4940

The 200 realization identifiers are shuffled once with a split seed of 2024 and partitioned into 120 source-training, 20 source-validation, 20 old-test, 16 target-update, 4 target-validation, and 20 target-test realizations. Feature means and standard deviations are fitted exclusively to source-training realizations. The same scaler is applied to every subsequent partition. All windows from one realization remain in one partition, so directly overlapping windows do not cross development and test sets, and target-domain information does not enter source-model fitting. Because the original source arrays are publicly shared, readers can independently inspect the positive-reset reconstruction rule and verify that the realization grouping precedes window-level partitioning.

3.3. Controlled Sensor-Distribution Shift

The old domain consists of normalized source-distribution windows. The new domain retains the same realizations and TBF target definition but applies a controlled shift in the source-training standardized feature space. If z is a standardized window, the shifted window z' is

$$z' = g \odot z + b + \epsilon, \tag{1}$$

where each channel gain is sampled from $g_j \in [0.95, 1.05]$, the signed channel bias has magnitude $|b_j| \in [0.10, 0.30]$, and $\epsilon \sim \mathcal{N}(0, 0.10^2)$. Gain, signed bias, and the two dropout-channel identities are sampled once per realization (independently across channels where applicable). Gaussian noise is sampled independently for each window, time point, and

channel. For each window, the dropout start is sampled independently and a contiguous block of 15 time points (5% of 300) is set to zero in the realization's two selected channels. Every draw is reproducible under a shift seed of 7419.

This construction isolates sensor-distribution robustness under controlled bias, gain, noise, and dropout. It does not encode a verified mapping to weather, payload, flight-control modes, hardware aging, or maintenance interventions.

3.4. Candidate Architectures

Three sequence regressors are evaluated under the same input and TBF target:

- **CNN–LSTM**: Four 1D convolutional blocks with 64, 128, 256, and 512 channels feed a single-layer LSTM with 1024 hidden units, which is followed by a 128-unit regression head and dropout 0.5.
- **TCN–CNN**: Four residual temporal-convolution blocks use dilation rates 1, 2, 4, and 8, which are followed by two convolutional layers, global pooling, and a regression head.
- **Transformer–CNN**: A one-layer, two-head Transformer encoder operates on the 10-dimensional sequence with sinusoidal positional encoding, followed by four convolutional stages and global pooling.

Each architecture is trained on source-training realizations for five epochs with Adam, a learning rate of 10^{-3}, a weight decay of 10^{-5}, gradient clipping at 1.0, and seeds 2024, 2025, and 2026. The epoch with the lowest source-validation RMSE is saved. For every subsystem, the architecture with the lowest mean source-validation RMSE over the three seeds is selected before incremental evaluation.

Figure 1 summarizes the implemented feature flow and layer dimensions used in the source-validation comparison.

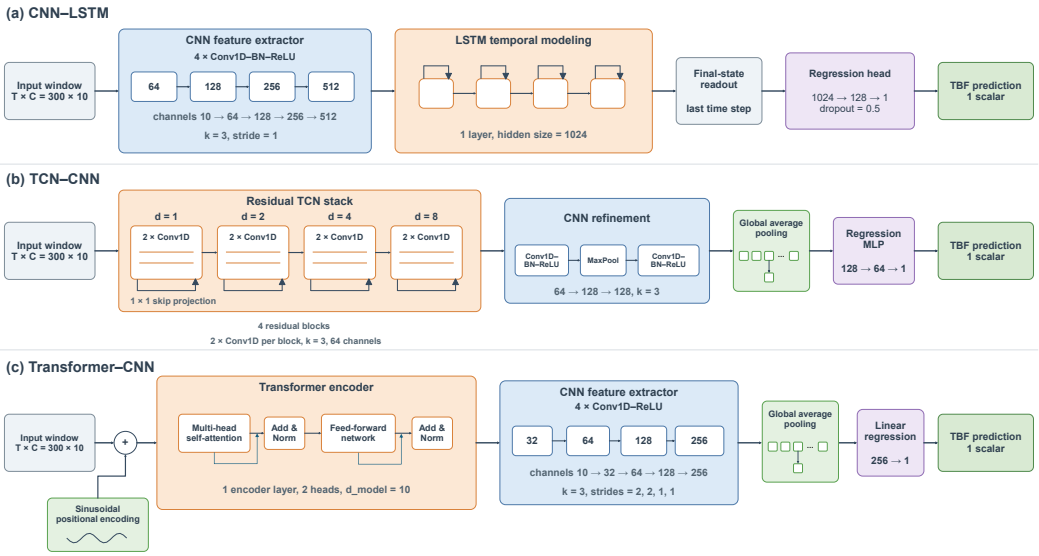


Figure 1. Candidate sequence-regression architectures evaluated with the same 300 × 10 input window and scalar TBF target. (**a**) CNN–LSTM uses four convolutional feature-extraction stages and a one-layer LSTM. (**b**) TCN–CNN uses four residual temporal-convolution blocks followed by convolutional pooling. (**c**) Transformer–CNN uses sinusoidal positional encoding, a one-layer Transformer encoder, and four convolutional stages. Dimensions and hyperparameters correspond to the implementations used in the source-validation comparison.

3.5. Incremental Update Strategies

Let θ_0 denote the selected source-model parameters and θ the updated parameters. Fine tuning minimizes the target-update MSE,

$$\mathcal{L}_{\text{task}}(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f_\theta(x_i))^2. \tag{2}$$

The proposed replay-free update adds an L2 drift constraint,

$$\mathcal{L}_{\text{L2}}(\theta) = \mathcal{L}_{\text{task}}(\theta) + \lambda \sum_p \|\theta_p - \theta_{0,p}\|_2^2. \tag{3}$$

The benchmark contains six strategies under the same realization partitions and a five-epoch update budget:

1. **No update**: Evaluate the source model directly on old and shifted domains.
2. **Fine tuning**: Optimize the target-update MSE without a preservation term.
3. **Selected L2**: Apply Equation (3) with a validation-selected parameter-drift weight.
4. **EWC**: Penalize parameter changes using an empirical Fisher diagonal estimated from eight source-training batches and a validation-selected penalty weight.
5. **Replay**: Mix the target-update set with a validation-selected fraction of source-training realizations sampled deterministically for each seed.
6. **LwF**: Add the MSE between updated and source-model predictions on target-update samples using a validation-selected distillation weight.

The update optimizer is Adam with a weight decay of 10^{-5}, gradient clipping at 1.0, and batch size of 48 for the selected CNN–LSTM. At every epoch, both the source validation and shifted target validation are evaluated. The saved update checkpoint minimizes

$$S_{\text{val}} = \text{RMSE}_{\text{old,val}} + \text{RMSE}_{\text{new,val}}. \tag{4}$$

3.6. Budget-Matched Validation-Only Hyperparameter Selection

Each trainable strategy receives four predeclared candidates and the same five-epoch budget per candidate. Fine tuning searches learning rates $\{10^{-4}, 3\times10^{-4}, 10^{-3}, 3\times10^{-3}\}$. L2, EWC, and LwF search their respective penalty weights over $\{10^{-4}, 10^{-3}, 10^{-1}, 1\}$ at a learning rate of 10^{-3}. Replay searches source-realization fractions $\{0.05, 0.10, 0.20, 0.40\}$ at a learning rate of 10^{-3}. For each subsystem, seed, method, and candidate, the epoch minimizing Equation (4) is retained. Candidate scores are averaged over seeds, and the lowest mean fixes that method's candidate for held-out evaluation. Thus, every trainable baseline receives the same number of candidates, epochs, seeds, data partitions, validation objective, and prohibition on test-set access. No update has no tunable hyperparameter and is evaluated directly.

3.7. Evaluation Metrics

Global evaluation reports MSE, RMSE, MAE, bias, and R^2. Stability and plasticity are measured relative to the no-update model under the same subsystem and seed:

$$F_{\text{old}} = \text{MSE}_{\text{old,test}}^{\text{updated}} - \text{MSE}_{\text{old,test}}^{\text{no update}}, \tag{5}$$

$$G_{\text{new}} = \text{MSE}_{\text{new,test}}^{\text{no update}} - \text{MSE}_{\text{new,test}}^{\text{updated}}. \tag{6}$$

A lower F_{old} value indicates greater source retention; a higher G_{new} value indicates stronger target adaptation.

Checkpoint parameter drift is reported as the relative trainable-parameter distance $D_\theta = \|\theta - \theta_0\|_2 / \|\theta_0\|_2$. Batch-normalization running statistics are buffers and are excluded from this norm. A lower D_θ denotes a more conservative update from the selected source checkpoint.

Near failure is defined as $y \leq 2$ h. In addition to the window-level MAE, predictions are grouped by held-out realization to show the distribution of near-failure error. For a maintenance threshold $h \in \{1, 2, 3\}$ h, a missed warning occurs when $y \leq h$ and $\hat{y} > h$; a false warning occurs when $y > h$ and $\hat{y} \leq h$. Rates are computed conditionally over actual-positive and actual-negative windows, respectively. All reported uncertainty is mean $\pm$ sample standard deviation over three seeds. With three seeds, the analysis does not attach inferential significance claims to between-method differences.

3.8. Computational Budget and Reproducibility

All experiments were conducted under a fixed offline computational budget. The benchmark is designed for reproducible algorithm comparison and model-update analysis rather than onboard real-time deployment.

For architecture selection, all candidate networks were evaluated using the same training epochs, optimizer settings, random seeds, and validation criteria. For incremental learning evaluation, all trainable strategies received the same number of hyperparameter candidates, update epochs, seeds, data partitions, and validation objectives.

Therefore, the benchmark evaluates the stability–plasticity trade-off among learning strategies under a controlled computational budget rather than the embedded implementation feasibility.

4. Results

4.1. Architecture Selection

Figure 2 and Table 2 show the source-validation comparison. CNN–LSTM has the lowest mean validation RMSE in all three available TBF subsystems. The airframe comparison between CNN–LSTM and TCN–CNN is close, and the selection rule resolves it using the prespecified mean-validation criterion rather than test performance.

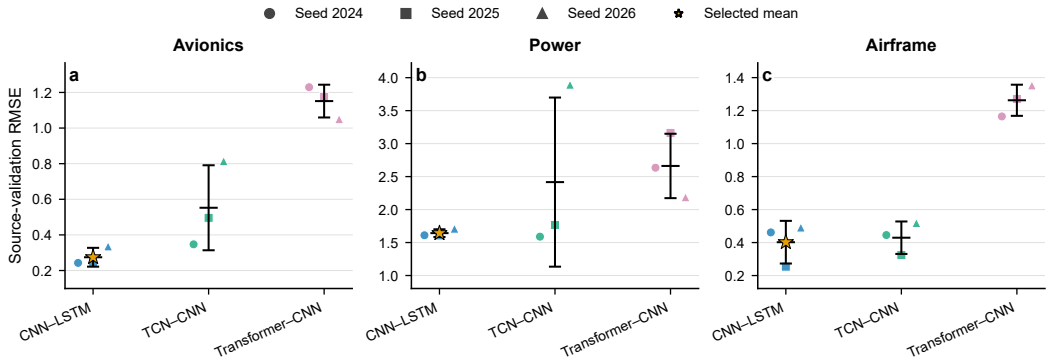


Figure 2. Source-validation architecture comparison. Panels (**a**–**c**) show avionics, power, and airframe; colors identify architectures and marker shapes identify seeds. Black ticks and error bars show mean $\pm$ sample SD ($n = 3$). The star marks the architecture selected by the lowest mean source-validation RMSE.

Table 2. Source-validation RMSE, mean $\pm$ sample SD over three seeds. Bold values identify the selected architecture.

Subsystem	CNN–LSTM	TCN–CNN	Transformer–CNN
Avionics	**0.2745 $\pm$ 0.0529**	0.5524 $\pm$ 0.2387	1.1515 $\pm$ 0.0920
Power	**1.6427 $\pm$ 0.0584**	2.4163 $\pm$ 1.2821	2.6614 $\pm$ 0.4881
Airframe	**0.4022 $\pm$ 0.1293**	0.4295 $\pm$ 0.0984	1.2627 $\pm$ 0.0944

4.2. Budget-Matched Method-Specific Hyperparameter Selection

Figure 3 applies the same validation-only decision rule to all five trainable methods. Each method has four candidates and 12 validation observations per subsystem (four candidates × three seeds). The selected candidates in Table 3 differ by both method and subsystem. Replay obtains the lowest mean validation score for power and airframe, whereas fine tuning obtains the lowest score for avionics. This result motivates method-specific tuning rather than treating one baseline as fully tuned while fixing the others.

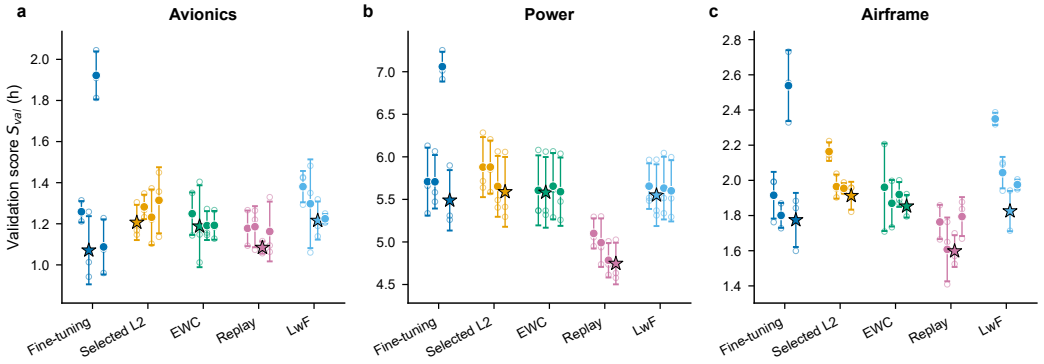


Figure 3. Budget-matched validation-only hyperparameter selection. Panels (**a**–**c**) show avionics, power, and airframe; colors identify methods, including the selected stars. Open symbols show the three seed scores for each of four candidates; filled symbols and error bars show mean ± sample SD. Stars identify the candidate selected for held-out testing. Candidate order follows the grids stated in Section 3.6.

Table 3. Validation-selected candidate and score, mean ± sample SD over three seeds.

Subsystem	Method	Selected Candidate	S_{val}
Avionics	Fine tuning	LR 10^{-4}	1.071 ± 0.166
Avionics	L2	penalty 1	1.206 ± 0.086
Avionics	EWC	penalty 10^{-1}	1.188 ± 0.199
Avionics	LwF	penalty 10^{-3}	1.216 ± 0.093
Avionics	Replay	fraction 0.20	1.084 ± 0.030
Power	Fine tuning	LR 3×10^{-4}	5.489 ± 0.355
Power	L2	penalty 10^{-4}	5.588 ± 0.410
Power	EWC	penalty 10^{-1}	5.582 ± 0.415
Power	LwF	penalty 10^{-1}	5.549 ± 0.363
Power	Replay	fraction 0.40	4.746 ± 0.244
Airframe	Fine tuning	LR 3×10^{-4}	1.775 ± 0.154
Airframe	L2	penalty 10^{-4}	1.913 ± 0.079
Airframe	EWC	penalty 10^{-4}	1.853 ± 0.064
Airframe	LwF	penalty 10^{-3}	1.825 ± 0.114
Airframe	Replay	fraction 0.20	1.598 ± 0.090

4.3. Held-Out Stability and Plasticity

Table 4 and Figure 4 report the held-out domain results and corresponding stability–plasticity distributions after selection. Every update strategy produces positive new-domain adaptation gain relative to no update. In avionics, tuned fine tuning improves both old-domain MSE and new-domain MSE relative to no update with $F_{\text{old}} = -0.021$ and $G_{\text{new}} = 1.107$. In power and airframe, replay occupies the strongest retention–adaptation region with negative mean forgetting (-0.280 and -0.029) and gains of 5.090 and 2.351,

respectively. The tuned L2 update remains effective on the shifted domain, but it does not dominate tuned fine tuning or replay under the budget-matched protocol.

Table 4. Held-out comparison, mean ± sample SD over three seeds. F_{old} is old-domain forgetting in MSE; G_{new} is new-domain adaptation gain in MSE.

Subsystem	Method	Old-Test RMSE	New-Test RMSE	F_{old}	G_{new}	New Near-Failure MAE
Avionics	No update	0.276 ± 0.054	1.453 ± 0.017	0.000 ± 0.000	0.000 ± 0.000	0.483 ± 0.034
Avionics	Fine tuning	0.238 ± 0.017	1.000 ± 0.097	−0.021 ± 0.025	1.107 ± 0.221	0.316 ± 0.034
Avionics	Selected L2	0.308 ± 0.015	1.155 ± 0.078	0.017 ± 0.023	0.775 ± 0.223	0.374 ± 0.048
Avionics	EWC	0.347 ± 0.104	0.991 ± 0.100	0.049 ± 0.093	1.124 ± 0.200	0.294 ± 0.043
Avionics	LwF	0.379 ± 0.042	1.008 ± 0.019	0.067 ± 0.062	1.096 ± 0.079	0.280 ± 0.037
Avionics	Replay	0.277 ± 0.036	1.050 ± 0.059	0.000 ± 0.028	1.008 ± 0.147	0.279 ± 0.005
Power	No update	1.866 ± 0.047	4.079 ± 0.251	0.000 ± 0.000	0.000 ± 0.000	4.201 ± 0.437
Power	Fine tuning	2.314 ± 0.234	3.406 ± 0.154	1.909 ± 0.936	5.069 ± 1.019	3.675 ± 0.240
Power	Selected L2	2.487 ± 0.221	3.399 ± 0.094	2.733 ± 0.950	5.123 ± 1.472	3.594 ± 0.241
Power	EWC	2.447 ± 0.257	3.396 ± 0.076	2.548 ± 1.122	5.149 ± 1.557	3.607 ± 0.251
Power	LwF	2.439 ± 0.204	3.382 ± 0.103	2.491 ± 0.845	5.237 ± 1.380	3.701 ± 0.220
Power	Replay	1.789 ± 0.051	3.403 ± 0.156	−0.280 ± 0.047	5.090 ± 1.001	3.215 ± 0.289
Airframe	No update	0.400 ± 0.127	1.837 ± 0.263	0.000 ± 0.000	0.000 ± 0.000	0.740 ± 0.067
Airframe	Fine tuning	0.444 ± 0.066	1.175 ± 0.113	0.030 ± 0.039	2.030 ± 0.987	0.502 ± 0.058
Airframe	Selected L2	0.527 ± 0.104	1.241 ± 0.113	0.114 ± 0.017	1.871 ± 1.019	0.499 ± 0.055
Airframe	EWC	0.579 ± 0.077	1.208 ± 0.074	0.169 ± 0.025	1.958 ± 0.876	0.516 ± 0.065
Airframe	LwF	0.551 ± 0.077	1.215 ± 0.086	0.137 ± 0.033	1.938 ± 0.905	0.505 ± 0.084
Airframe	Replay	0.374 ± 0.048	1.032 ± 0.078	−0.029 ± 0.073	2.351 ± 0.962	0.372 ± 0.059

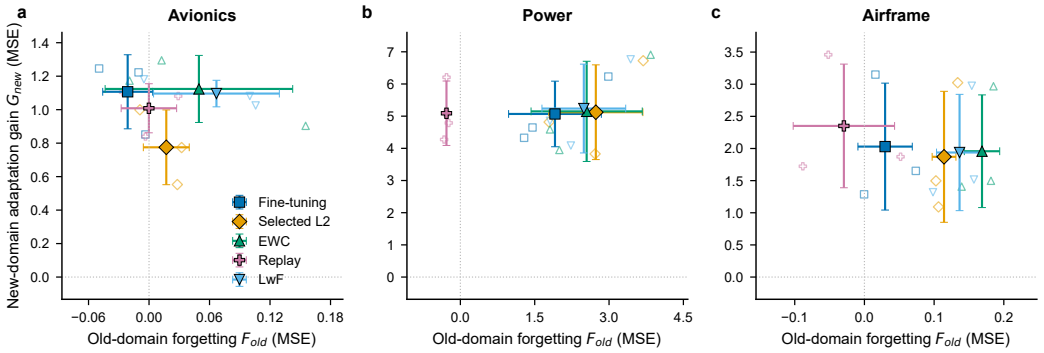


Figure 4. Held-out stability–plasticity comparison. Panels (**a**–**c**) show avionics, power, and airframe; colors and markers identify methods. Faint symbols show individual seeds; emphasized symbols and bidirectional error bars show mean ± sample SD ($n = 3$). The preferred direction is toward lower old-domain forgetting and higher new-domain adaptation gain.

Replay retains a validation-selected fraction of source realizations, whereas selected L2, EWC, and LwF do not retain source windows during the update. The comparison therefore distinguishes two operating conditions: replay when source storage is permitted, and regularized updating when the update must remain replay-free.

4.4. Near-Failure and Threshold-Oriented Evaluation

Every selected update reduces new-domain near-failure MAE relative to no update (Figure 5). Replay gives the lowest means for power and airframe (3.215 and 0.372 h), while replay and LwF are nearly tied for avionics (0.279 and 0.280 h). The L2 means are 0.374, 3.594, and 0.499 h for avionics, power, and airframe, respectively. These rankings reinforce that the appropriate update rule depends on the subsystem and whether source storage is permitted.

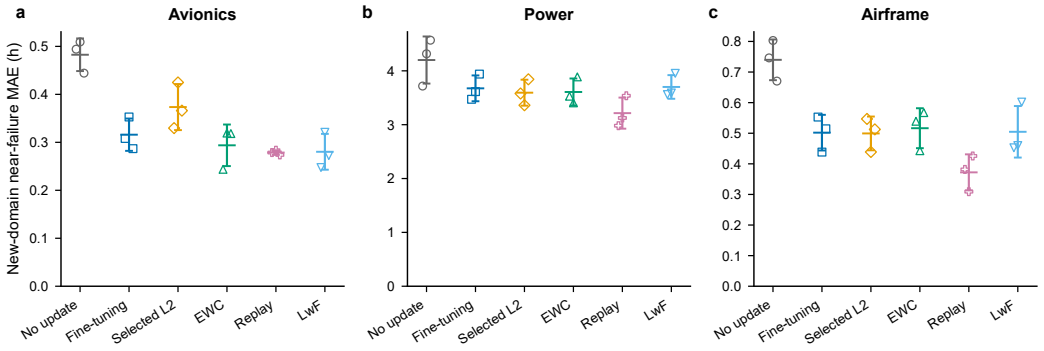


Figure 5. New-domain near-failure MAE for TBF $\leq$ 2 h. Panels (**a**–**c**) show avionics, power, and airframe; colors and markers identify methods. Individual symbols show seeds; colored symbols and horizontal error bars show mean $\pm$ sample SD ($n = 3$).

Figure 6 converts the regression output into maintenance-threshold decisions. For the selected L2, the missed-warning rates at 1, 2, and 3 h are 46.1%, 18.6%, and 15.7% for avionics; 99.1%, 87.9%, and 72.3% for power; and 35.5%, 20.7%, and 10.1% for airframe. The corresponding false-warning rates are 0.3%, 1.0%, and 1.7%; 0.0%, 1.2%, and 4.8%; and 1.2%, 3.5%, and 5.0%, respectively. Across methods, power missed-warning rates remain 90.4–99.8% at 1 h and 65.5–75.3% at 3 h. The power case therefore does not satisfy a high-sensitivity maintenance gate at the evaluated thresholds even though its global regression error improves relative to no update. This threshold analysis prevents the aggregate adaptation gain from being interpreted as operational readiness.

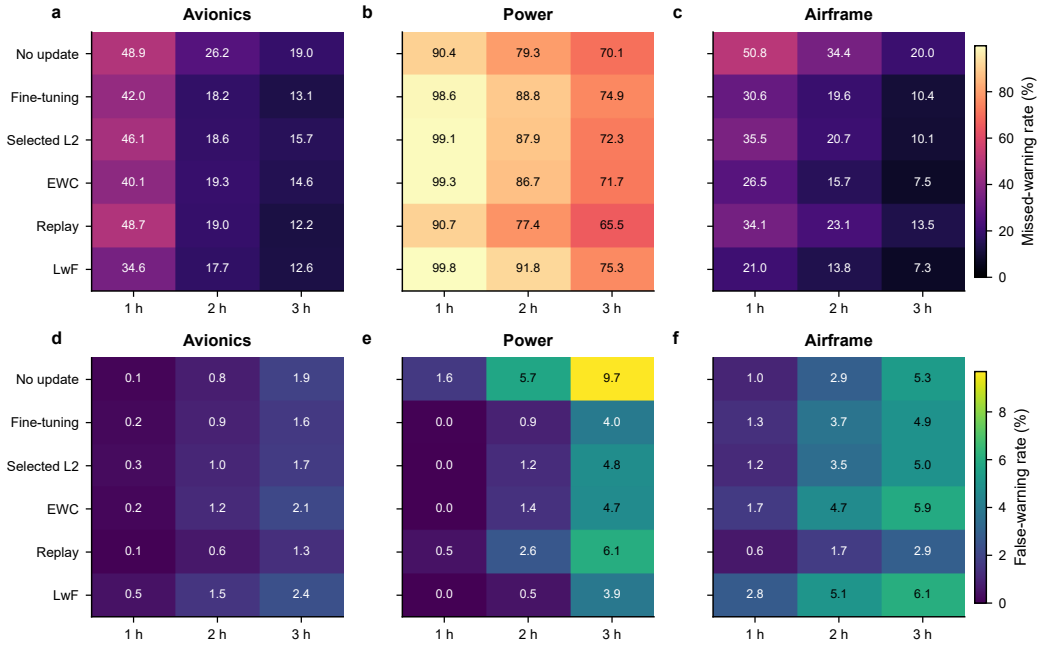


Figure 6. Threshold-based warning rates on the held-out new domain. Cells report mean rates (%) over three seeds for 1, 2, and 3 h TBF thresholds. Panels (**a**–**c**) show missed-warning rates; panels (**d**–**f**) show false-warning rates. Separate perceptually uniform scales are used because the two rate types occupy different numerical ranges.

5. Discussion

5.1. What the Constraint Contributes

The central result of the budget-matched benchmark is that the stability–plasticity choice is method and subsystem dependent. Tuning fine tuning itself changes the comparison: it gives the lowest mean forgetting in avionics and lower forgetting than the three

replay-free regularizers in power and airframe. L2 provides a controlled replay-free update and positive new-domain gain in all subsystems, but the evidence does not support a universal advantage over tuned fine tuning, EWC, or LwF.

Replay provides a strong reference when historical data can be retained. Its power and airframe points combine negative mean forgetting with substantial adaptation gain. L2 addresses a different operational condition: historical windows are not used during the update, and the stored source model itself provides the reference. EWC shares the replay-free property but requires a Fisher approximation; LwF requires teacher inference on the update data. Direct L2 regularization is therefore retained as the simplest replay-free preservation mechanism among the evaluated strategies rather than positioned as the overall winner.

5.2. Maintenance-Oriented Acceptance

Global regression metrics and threshold decisions answer different questions. A positive adaptation gain confirms that the shifted-domain MSE is lower than no update. A missed-warning rate tests whether that improvement is sufficient at a maintenance boundary. The power results demonstrate why both are required: the incremental methods reduce new-domain error, but the 1–3 h missed-warning rates remain too high for a safety-critical warning role. A deployment process should therefore use subsystem-specific acceptance gates that include old-domain retention, new-domain regression error, near-failure error, and missed-warning limits.

5.3. Evidence Scope and Limitations

The evidence is based on publicly shared RflySim-derived simulation records and offline batch updates. Real UAV data can include flight-control mode changes, weather, payload variation, vibration coupling, sensor aging, maintenance interventions, electromagnetic interference, and hardware-specific noise. The bias, gain, noise, and dropout perturbations used here do not reproduce those physical mechanisms. The results establish algorithm behavior under a controlled sensor-distribution shift and do not establish flight readiness.

The source package supports three TBF subsystems and ten numeric channels per subsystem; it does not provide a verified semantic channel dictionary or an electromechanical TBF array. The revised scope follows the available evidence and excludes the earlier electromechanical result. This paper reports split manifests, shift parameters, training seeds, and tensor geometry so that the implemented benchmark can be reproduced without inventing unavailable physical metadata.

Repeated seeds quantify optimization variability, while independent physical simulation campaigns would be needed to quantify uncertainty across vehicle configurations and degradation mechanisms. The reported error bars therefore describe optimization-seed variability under the fixed controlled simulation protocol; they do not quantify physical fleet variability.

The next validation stage should retain the same old/new protocol while replacing the controlled perturbation with bench, HIL, or flight trajectories. That stage should predefine missed-warning limits for each subsystem and include model rollback when either source retention or near-failure sensitivity violates the acceptance criterion.

6. Conclusions

This paper established a leakage-controlled, validation-selected benchmark for incremental UAV TBF regression on simulated avionics, power, and airframe data. Source-validation selection chose CNN–LSTM for all three subsystems. A budget-matched protocol

then gave fine tuning, L2, EWC, LwF, and replay four candidates each under identical seeds, update epochs, partitions, and old-plus-new validation scoring.

On held-out realizations, tuned fine tuning gave the lowest mean forgetting in avionics, while replay gave the strongest retention–adaptation balance in power and airframe when source storage was permitted. L2 adapted to the shifted domain without retaining source windows and remains a simple replay-free option, but no single update method dominated across all subsystems and operating constraints. Near-failure and threshold metrics added a maintenance decision layer and identified the power subsystem as requiring a higher-sensitivity model or warning policy before operational consideration.

The resulting contribution is an auditable, budget-matched selection protocol with direct forgetting measurement and maintenance-oriented evaluation under a controlled simulation shift. The released dataset and associated metadata provide a reproducible basis for future extensions toward bench, hardware-in-the-loop, and flight-data validation. Physical UAV validation remains the required step for claims beyond simulation feasibility.

Author Contributions: Conceptualization, G.R., X.M. and J.W. (Jian Wang); methodology, G.R. and J.W. (Jian Wang); software, G.R., D.W. and C.T.; validation, G.R., D.W. and Y.H.; formal analysis, G.R. and C.T.; investigation, G.R., D.W., C.T. and Y.H.; resources, X.M., J.W. (Jian Wang) and J.W. (Jianyuan Wang); data curation, G.R. and Y.H.; writing—original draft preparation, G.R.; writing—review and editing, D.W., C.T., Y.H., X.M., J.W. (Jian Wang) and J.W. (Jianyuan Wang); visualization, G.R. and C.T.; supervision, X.M. and J.W. (Jianyuan Wang); project administration, X.M., J.W. (Jian Wang) and J.W. (Jianyuan Wang); funding acquisition, J.W. (Jian Wang). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Postgraduate Research & Practice Innovation Program of NUAA (Grant No. xcxjh20251507).

Institutional Review Board Statement: Not applicable. This paper features simulated UAV subsystem data and did not involve humans or animals.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original simulation dataset used in this paper is publicly available at https://doi.org/10.5281/zenodo.22058477. The shared archive contains the source arrays used for the avionics, power, and airframe-structure TBF experiments reported in this paper. The realization split, run manifest, aggregate result records, analysis scripts, and trained-model identifiers generated for this revision are available from the corresponding author on reasonable request. The shared records were generated from a customized RflySim-based simulation configuration and are not the public RflySimSafe ST001–ST029 dataset. The complete executable customized RflySim project and platform-version manifest are not included in the shared archive.

Conflicts of Interest: Author Di Wu was employed by China Rong Tong Academy of Sciences Group Corporation Limited. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Jardine, A.K.S.; Lin, D.; Banjevic, D. A Review on Machinery Diagnostics and Prognostics Implementing Condition-Based Maintenance. *Mech. Syst. Signal Process.* **2006**, *20*, 1483–1510. [CrossRef]
2. Si, X.S.; Wang, W.; Hu, C.H.; Zhou, D.H. Remaining Useful Life Estimation: A Review on the Statistical Data Driven Approaches. *Eur. J. Oper. Res.* **2011**, *213*, 1–14. [CrossRef]
3. Lei, Y.; Li, N.; Guo, L.; Li, N.; Yan, T.; Lin, J. Machinery Health Prognostics: A Systematic Review from Data Acquisition to RUL Prediction. *Mech. Syst. Signal Process.* **2018**, *104*, 799–834. [CrossRef]
4. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
5. Bai, S.; Kolter, J.Z.; Koltun, V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv* **2018**, arXiv:1803.01271.

6. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 5998–6008.
7. Zhao, R.; Yan, R.; Chen, Z.; Mao, K.; Wang, P.; Gao, R.X. Deep Learning and Its Applications to Machine Health Monitoring. *Mech. Syst. Signal Process.* **2019**, *115*, 213–237. [CrossRef]
8. Parisi, G.I.; Kemker, R.; Part, J.L.; Kanan, C.; Wermter, S. Continual Lifelong Learning with Neural Networks: A Review. *Neural Netw.* **2019**, *113*, 54–71. [CrossRef]
9. De Lange, M.; Aljundi, R.; Masana, M.; Parisot, S.; Jia, X.; Leonardis, A.; Slabaugh, G.; Tuytelaars, T. A Continual Learning Survey: Defying Forgetting in Classification Tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 3366–3385. [CrossRef]
10. Rebuffi, S.A.; Kolesnikov, A.; Sperl, G.; Lampert, C.H. ICaRL: Incremental Classifier and Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; IEEE: Piscataway, NJ, USA, 2017; pp. 2001–2010.
11. Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A.A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. Overcoming Catastrophic Forgetting in Neural Networks. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 3521–3526. [CrossRef]
12. Li, Z.; Hoiem, D. Learning without Forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 2935–2947. [CrossRef]
13. Lopez-Paz, D.; Ranzato, M. Gradient Episodic Memory for Continual Learning. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 6467–6476.
14. Jiao, Y.; Jiang, H.; Wang, X.; Liu, Y.; Qiao, Y. A Wiener-Prior-Guided Hierarchical Network for Aero-Engine RUL Prediction with Uncertainty Quantification. *Meas. Sci. Technol.* **2026**, *37*, 275106. [CrossRef]
15. Wang, X.; Jiang, H.; Dong, Y.; Mu, M. Spatial-Channel Collaborative Multi-Scale Graph Interaction Deep Transfer Learning for Unsupervised Rotating Machinery Fault Diagnosis. *Eng. Appl. Artif. Intell.* **2026**, *176*, 114691. [CrossRef]
16. Dong, Y.; Jiang, H.; Wang, X.; Li, Z. Entropy-Oriented Semi-Supervised Dynamic Prototype Contrastive Learning for Rotating Machinery Fault Diagnosis. *IEEE/ASME Trans. Mechatron.* **2025**, *30*, 4934–4945. [CrossRef]
17. Li, Y.; He, J.; Zhou, D.H. A Review of Condition-Based Maintenance: Its Prognostic and Operational Aspects. *Front. Eng. Manag.* **2020**, *7*, 323–334. [CrossRef]
18. Bang, J.; Kim, H.; Yoo, Y.; Ha, J.W.; Choi, J. Rainbow Memory: Continual Learning with a Memory of Diverse Samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; IEEE: Piscataway, NJ, USA, 2021; pp. 8218–8227.
19. Rusu, A.A.; Rabinowitz, N.C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; Hadsell, R. Progressive Neural Networks. *arXiv* **2016**, arXiv:1606.04671.
20. Dai, X.; Ke, C.; Quan, Q.; Cai, K.Y. RFlySim: Automatic Test Platform for UAV Autopilot Systems with FPGA-Based Hardware-in-the-Loop Simulations. *Aerosp. Sci. Technol.* **2021**, *114*, 106727. [CrossRef]
21. Wang, S.; Dai, X.; Ke, C.; Quan, Q. RflySim: A Rapid Multicopter Development Platform for Education and Research Based on Pixhawk and MATLAB. In *Proceedings of the 2021 International Conference on Unmanned Aircraft Systems (ICUAS)*; IEEE: Piscataway, NJ, USA, 2021; pp. 1587–1594. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.