ANOMALY LOCALIZATION AS A DEFENSE AGAINST SENSOR FAULTS IN VIRTUAL SENSING

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning-based virtual sensors inherit the vulnerability of the physical sensors they consume. When a transducer drifts, saturates, or sticks, the input distribution shifts and predictions degrade. We tackle this with a detect–isolate–accommodate pipeline, in which a fault detector flags and discards failing sensors before they reach the predictor. This shifts the burden from learning invariance across the full fault manifold to recognizing departures from normality, collapsing the predictor's necessary training augmentation to plain channel dropout. To make this pipeline work, what matters is not only *when* a channel misbehaves but *which* one does. Yet time-series anomaly detectors are built and benchmarked almost exclusively for the former. We test whether they can serve as channel-level isolators end-to-end on a benchmark for sensor-failure robustness in virtual sensing, spanning six domains, nine datasets and ten failure modes. Finding that current state-of-the-art detectors localize faulty channels poorly, we propose ITX, an iTransformer adaptation with untied per-channel embeddings and projections, trained with cross-instance channel substitution to simulate plausible-but-wrong sensor readings. It substantially improves channel-level localization and, paired with a dropout-trained back-end, makes explicit fault isolation the best-ranked robustification in our evaluation.

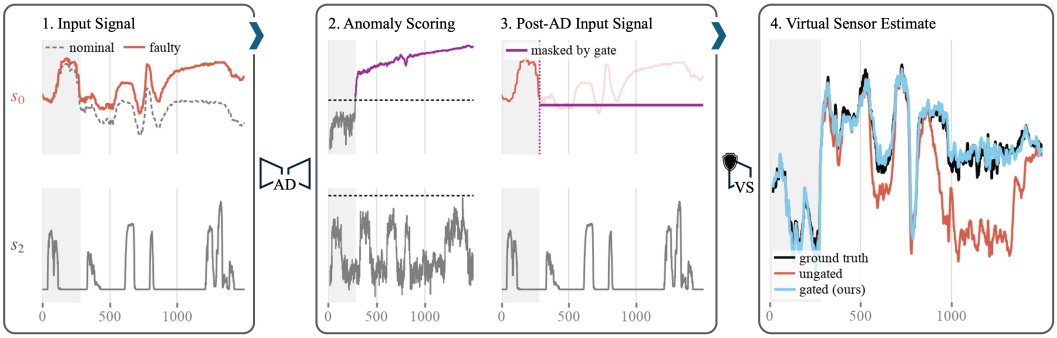


Figure 1: **Motivating example.** A drift in one input sensor that gets ingested by a sideslip angle virtual sensor can cause huge deviations in the output. Properly localizing the fault and masking it improves robustness significantly.

1 INTRODUCTION

Virtual sensing, the inference of a hard-to-measure quantity from co-located, readily available measurements, is a core enabler of monitoring and closed-loop control in cyber-physical systems (Martin et al., 2021). Where faithful physical models are unavailable, this inference is increasingly performed by purely data-driven predictors (Brandt et al., 2026a). A defining property of the deployed setting is that the target is unavailable by construction. If it could be measured online, no virtual sensor would be required. The predictor therefore has no residual with which to monitor itself, and every safeguard against degradation must act on the *input* side. This is especially consequential in closed-loop or autonomous deployments, where the estimate drives an actuator directly and no

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

human is positioned to catch a bad reading before it does harm. The most consequential input-side disturbance in this setting is sensor failure. Sensing hardware drifts, saturates, and sticks (Balaban et al., 2009; Jesus et al., 2017).

Following Jesus et al. (2017), we understand a sensor failure as arising at the sensing interface itself, which is what makes the resulting archetypes general across sensor types and application domains rather than tied to one physical failure mechanism. Formally, a failure on channel c replaces the true input x^{true} with a corrupted observation $\tilde{x}$, inducing a covariate shift $p(\tilde{x}) \neq p(x)$ under an unchanged $p(y \mid x^{\text{true}})$ for the target y, so the learned predictor f_θ is no longer the mapping required at test time. In practice, this is severe. On MuViS-C (Brandt et al., 2026b), a virtual sensing benchmark for robustness under sensor faults, every evaluated architecture becomes worse than a naive mean predictor on at least one failure-channel combination.

Defenses against such sensor failures fall into three families. (i) *Model-side invariance* prepares the predictor for corrupted inputs in advance, through augmentation (Mena et al., 2024), adversarial training (Madry et al., 2018), or fault-aware self-supervised pretraining (Brandt et al., 2025). (ii) *Repair* leaves the predictor unchanged and instead corrects the input as it arrives, substituting a plausible clean value via denoising autoencoders (Vincent et al., 2008) or diffusion-based purification (Nie et al., 2022). (iii) *DIA*, named after the control community's separation of fault *detection*, *isolation*, and *accommodation* as distinct sub-problems (Isermann, 2006; Blanke et al., 2015), instead localizes the corrupted channel and withholds it, rather than substituting a value. The resulting missing information must then be accommodated by the downstream system, for instance by falling back on redundant hardware, replanning around the loss, or, as we pursue here, a predictor explicitly trained to handle missing channels.

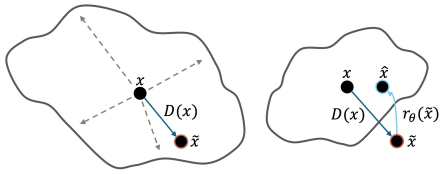


Figure 2: Model-side *invariance* widens the training distribution (left). *Repair* pushes faulty signals back toward the learned distribution (right). *DIA* pipelines instead combine a front-end that isolates faulty channels with a back-end that accommodates their absence.

We argue that DIA is the most robust of the three families, for two reasons. First, we conjecture an asymmetry in difficulty, grounded in the general case for anomaly detection: while the nominal region can be bounded, the space of possible deviations may be unbounded. This motivates learning a model of normality and scoring departures from it, rather than explicitly modeling the deviations themselves (Ruff et al., 2021). Invariance and repair take the harder route by construction, anticipating the entire fault manifold spanned by mode, channel, and severity in advance. Hardening against one corruption type can also transfer poorly to others (Geirhos et al., 2020). Detection only needs to answer a narrower question, whether a channel currently looks normal. Sensor readings arrive as multivariate time series, and answering that question for each channel over time is precisely the task time-series anomaly detection is built for. We therefore instantiate the detection step of DIA by attaching a time-series anomaly detector (TSAD) in front of the predictor.

Second, accommodation can borrow the machinery of model-side invariance (i) while drastically shrinking what the predictor must be invariant to, avoiding the hardware redundancy or per-subset models that other accommodation strategies require. Rather than anticipating a catalogue of fault-specific corruptions, the predictor only needs to tolerate missing channels. We realize this with a single, mode-agnostic augmentation, random input-channel dropout, so that one predictor tolerates any subset of missing channels rather than the full fault manifold. The available evidence for this asymmetry is encouraging. Given an *oracle* fault detector, a dropout-trained predictor degrades only marginally under corruption, while the same undefended predictor degrades substantially (Brandt et al., 2026b). That result, however, presumes perfect channel-level isolation; whether practical anomaly detectors can localize accurately enough for DIA to remain effective is still untested.

We answer this by instantiating DIA with three recent time-series anomaly detectors as the front-end and an input-dropout-trained ModernTCN as the back-end, and evaluate the composed pipeline end-to-end on MuViS-C, across six domains, nine datasets and ten severity-parameterized failure modes. Gated by an oracle detector, DIA is by far the strongest robustification on the benchmark, confirming the potential of the defense. Real detectors fall short of it. Mainly designed and scored to detect anomalies in *time*, they cannot localize them in *channel* accurately enough for the gate to pay off. We address this with ITX, a detector designed for channel-level localization.

Our contributions are:

- **An end-to-end evaluation of fault localization for robust virtual sensing.** We cast sensor-fault robustness in virtual sensing as a detect–isolate–accommodate problem, which reduces the predictor's burden from invariance over the full fault manifold to tolerating missing channels. Using MuViS-C, we evaluate whether TSAD methods can localize faulty channels, scoring them both directly and by the downstream effect of their decisions on a dropout-trained virtual sensor, bounded by ungated and oracle-gated references.
- **ITX: a detector built for localization.** An iTransformer adaptation with untied per-channel embeddings and projections, trained with cross-instance channel substitution to simulate plausible-but-wrong sensor readings, that localizes faulty channels far more reliably than existing detectors. Paired with a dropout-trained back-end, it closes most of the gap to the oracle and forms the best-ranked robustification in our evaluation.

2 Related Work

Robust virtual sensing. Virtual sensing research has largely optimized for nominal-condition accuracy, with robustness to sensor failure receiving comparatively little attention. MuViS consolidates virtual-sensing tasks from six domains into a unified benchmark, finding no dominant architecture under nominal conditions (Brandt et al., 2026a). Its extension, MuViS-C (Brandt et al., 2026b), adds ten realistic severity-parameterized sensor failure modes and evaluates non-robustified models alongside train-time hardening approaches, input dropout, adversarial training, and pretraining on synthetic sensor faults. All of these prepare the predictor itself rather than filtering its input. We close this gap by proposing a robustification strategy that combines an anomaly detector with an input-dropout-trained virtual sensor.

Time-series anomaly detection and localization. Recent detectors improve on reconstruction baselines through frequency patching and learned channel correlations (Wu et al., 2024), cross-scale association modeling (Li et al., 2025), and large-scale synthetic pretraining for zero-shot transfer (Lan et al., 2026). Evaluation remains centered on time-step detection over benchmarks whose anomalies are sparse and transient (Liu & Paparrizos, 2024), a regime that differs from persistent single-channel sensor faults both in statistics and in what the downstream system must be told. A smaller line of work targets anomaly *localization*, identifying which channels carry the anomaly, typically by attributing per-channel reconstruction error (Su et al., 2019; Dai et al., 2024), by tracing causal deviations (Han et al., 2024), or, most recently, by reweighting those errors with contribution weights derived from the model's learned projections (Shimillas et al., 2026). In the sensor-fault literature, Fu & Yan (2024) isolate faulty sensors via leave-one-channel-out reconstruction residuals, but repair rather than withhold them. In this paper we test whether modern TSAD detectors and localization methods can fill the detection and isolation roles under persistent, in-range single-channel faults, and whether the resulting isolation supports the robustification we propose.

3 Experimental Setup

3.1 Virtual Sensing Under Sensor Failures

A virtual sensor is a mapping $f_\theta : \mathbb{R}^{C\times T} \to \mathbb{R}$ from a window of C input channels observed over T time steps to a scalar estimate $\hat{y}$ of a target quantity at the reference time t_0 (Brandt et al., 2026a). Unlike forecasting, the output is anchored at the end of the context window and interpreted as a measurement a physical sensor could in principle have produced. Crucially, y is unavailable at deployment, so no residual $y - \hat{y}$ can be monitored.

We build on MuViS-C (Brandt et al., 2026b). Its corruption protocol acts on a single channel c through a fault function $D_s(\cdot)$ with severity $s \in [0, 1]$, producing $\tilde{x}^{(c)} = D_s(x^{(c)})$ while all remaining channels stay nominal. The benchmark's ten modes instantiate the failure archetypes documented by Balaban et al. (2009); Jesus et al. (2017): additive (bias, noise), multiplicative (constant and time-varying scaling), drift (linear, quadratic), saturation (hard and soft trimming), sporadic (outliers), and stuck-at (hard fault) corruptions. Evaluation instantiates D_s independently per (dataset, mode, victim channel, severity), corrupting one channel at a time; every combination is run

over the full test set, and Section 5 reports aggregates over this grid rather than single runs (more details in Appendix E). The exhaustive grid allows robustness to be analyzed along any of its axes, revealing, for example, whether a given defense holds up uniformly across failure types, or to trace how detection quality degrades as severity decreases toward the threshold where a fault is barely distinguishable from nominal noise, a regime of particular interest for anomaly detectors.

The benchmark covers nine datasets from six physical domains: particulate-matter estimation (Beijing PM2.5/PM10), heart-rate estimation from wrist-worn sensors (PPG-DaLiA), chemical concentration estimation (Tennessee Eastman Process), battery state-of-charge (Panasonic 18650PF), tire-temperature estimation (Vehicle Dynamics), and lateral velocity estimation across three motorsport sessions (REVS MMR/T66-13/T66-14). Channel counts range from $C{=}6$ to $C{=}33$ and context lengths from $T{=}20$ to $T{=}512$; per-dataset statistics are given in Table 5. For tractability, we cap the three largest test sets to their first 2,048 (HR), 12,288 (Battery) and 40,000 (TEP) windows when comparing detector gates; comparisons against the published MuViS-C baselines use full test sets.

3.2 The Detect–Isolate–Accommodate Pipeline

To defend f_θ against these faults, we decompose the defense into a front-end, responsible for detection and isolation, and a back-end, responsible for accommodation, consisting of a masking operator $\mathcal{A}_{\text{mask}}$ and the predictor f_θ (Figure 3). The front-end scores each entry of an incoming window from the output of a time-series anomaly detector h_ψ as $a(c,t)$, reduces these scores over time to $\bar{a}^{(c)}$ and, after thresholding at τ_c, returns a per-channel fault decision

$$\mathbf{d} \in \{0,1\}^C, \qquad d_c = 0 \iff \bar{a}^{(c)} > \tau_c, \tag{1}$$

where $d_c = 0$ declares channel c faulty. The decision is channel-level and window-wide: a channel declared faulty is no longer trusted and is discarded entirely for the duration of the window (Section 6 discusses finer-grained alternatives). The back-end accommodates the absence of the flagged channels. It applies the decision as a mask, zeroing them before prediction, which, under per-channel z-score normalization, is equivalent to replacing each by its training mean,

$$\check{x} = \mathcal{A}_{\text{mask}}(\tilde{x}, \mathbf{d}) = \mathbf{d} \odot \tilde{x} \in \mathbb{R}^{C\times T}, \qquad \hat{y} = f_\theta(\check{x}), \tag{2}$$

broadcasting $\mathbf{d}$ across the time dimension. Masked inputs never occur in nominal training data and would thus be out of distribution for a conventionally trained f_θ, so it is instead trained with random input-channel dropout, learning to predict from an arbitrary channel subset rather than only from the full array. This presumes redundancy in the input array, since f_θ must keep the target identifiable from the channels that survive the mask.

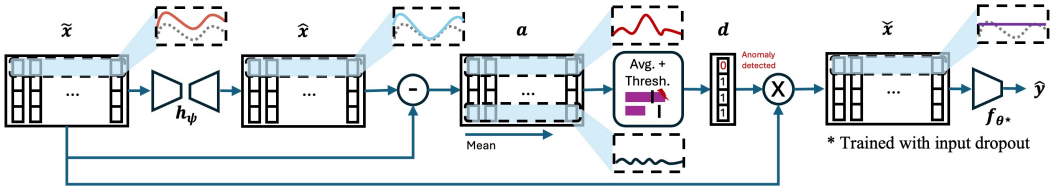


Figure 3: The detect–isolate–accommodate pipeline. The front-end returns a per-channel fault decision $\mathbf{d} \in \{0,1\}^C$ (0 = faulty), which the back-end applies as a mask. The virtual sensor f_θ, trained with random input-channel dropout, predicts from the surviving channels.

Front-end: detectors. We evaluate three recent detectors, plus a naive reference; the implementation details for all are given in Appendix F. **CATCH** (Wu et al., 2024) reconstructs patched frequency bands and learns patch-wise channel correlations through a channel-fusion module. **CrossAD** (Li et al., 2025) reconstructs fine-grained series from coarser ones and is channel-independent by construction, so no cross-channel information enters the score. **ALoRa** (Shimillas et al., 2026) constrains self-attention to be low-rank. Alongside **ALoRa-T**, that returns a detection score, it provides **ALoRa-Loc**, which attributes a detected anomaly to k channels using per-channel contribution weights derived from the model's learned projections. 3σ is a naive per-channel Shewhart reference (Chandola et al., 2009) that scores each sample by its distance from the channel's healthy mean in units of its healthy standard deviation, and flags a channel if any sample exceeds the limit. We introduce our own detector, ITX, separately in Section 4.

Back-end: virtual sensing. The back-end is a ModernTCN predictor (Donghao & Xue, 2023), trained in two variants: a *nominal* model trained on clean data only, and an ISENSD (Mena et al., 2024) variant, in which each input channel is independently zeroed with probability p=0.3 during training, preparing the model for the masking accommodation without exposing it to any specific fault mode. Architecture hyperparameters are taken from Brandt et al. (2026b).

3.3 EVALUATING CHANNEL-LEVEL ISOLATION

We first evaluate the front-end in isolation, using MuViS-C's corruption grid (Section 3.1), independent of the accommodation stage. Standard TSAD asks *when*, reducing per-(timestep, channel) scores along the channel axis. Isolation asks *which*, so we reduce the same scores along time instead, classifying each channel in a window independently, positive if that channel carries a fault. The baselines were built for the temporal task; we keep their scoring rules intact and change the aggregation axis.

Score reduction. Scores $a(c, t)$ are averaged over time to a window-level statistic $\bar{a}^{(c)}$ per channel (Equation 1). The exception is 3σ, which takes the maximum score over the channel's time steps.

Threshold calibration. Thresholds τ_c are calibrated per channel on $\bar{a}^{(c)}$ over clean validation windows at a target FPR of 1% to ensure stable operation during nominal conditions, including the 3σ limit, which is calibrated at that operating point rather than fixed at 3.

ALoRa-Loc. Its localization rule attributes an already-detected anomaly, so we run it as a cascade: windows clearing the 1% threshold enter stage two, which keeps a single channel among them. This emits exactly one channel per window, which matches our single-victim evaluation and therefore favours the method; a top-k rule would instead require a k that a detector does not know.

Metrics. We report precision and recall at that operating point, and top-1 localization accuracy: the fraction of corrupted windows whose highest-scoring channel is the victim. All metrics are stratified by failure mode and severity.

3.4 EVALUATING ROBUSTNESS AGAINST SENSOR FAILURES

Beyond isolation, we evaluate the full detect–isolate–accommodate pipeline end to end, following MuViS-C's original protocol on the same corruption grid (Section 3.1): each detector's decision now drives the masking back-end, and we record the accommodated virtual sensor's prediction RMSE. Each reported RMSE is a bootstrap mean over 200 resamples of the test set. We summarize the sweep with two complementary metrics.

mPC (mean Performance under Corruption): average RMSE across all corrupted conditions; in original units, lower is better.

rPC (relative Performance under Corruption): $\mathrm{rPC} = \mathrm{mPC} \,/\, \mathrm{RMSE}_{\mathrm{clean}}$; 1.0 = no degradation, values > 1 indicate performance loss.

The mPC metric captures absolute prediction quality under corruption; rPC isolates degradation from nominal accuracy, so a model with low mPC but high rPC is accurate yet fragile, while high mPC and low rPC signals a weak baseline that degrades gracefully. Cross-dataset aggregation uses geometric means over best-normalized values for mPC, direct geometric means for rPC, and mean ranks throughout, with Friedman tests and Nemenyi post-hoc analysis following Demšar (2006).

3.5 SYSTEM CONFIGURATIONS

For the end-to-end evaluation, each system pairs a back-end (nominal or ISENSD-trained) with a gate (none, detector, or ground-truth oracle). With the ISENSD back-end fixed, we compare **(a)** the six detector gates against **(b) Naive** (no gate) and **(c) Oracle**; the Naive–Oracle gap is the headroom available to a front-end, and the share a detector closes is its cost relative to perfect isolation. We further place ITX against **(d)** the MuViS-C pool, including its train-time hardened variants, on the full test sets. Repeating Naive, ITX and Oracle with the nominal back-end (Appendix K) tests whether gating and dropout training are complementary or substitutable. All gates are also scored on clean input (Table 13), where any mask is a false alarm.

4 ITX: CROSS-INSTANCE CHANNEL SUBSTITUTION FOR FAULT ISOLATION

To close the gap between current detectors and the oracle (Section 5.2), we introduce ITX, an iTransformer (Liu et al., 2024) adapted for fault isolation. A faulty sensor can stay within its physical range and remain plausible on its own (e.g., a constant offset on a temperature probe), so isolation must judge each channel against the rest of the array. iTransformer was proposed for forecasting; we keep its inverted encoder but train it to reconstruct. Beyond that, ITX differs from the standard reconstruction recipe in three ways. We train the model to reconstruct randomly selected channels from values swapped in from other samples in the batch, making cross-channel inference the objective itself; we give each channel its own embedding and projection head, since the array is fixed and channel identity is what isolation acts on; and we retain level and scale in the residual, unlike detectors that normalize on the way in and denormalize on the way out, which cancels a constant offset exactly. We ablate each of these choices in Appendix I.

Architecture. ITX follows the "inverted" layout of iTransformer (Liu et al., 2024): a window $x \in \mathbb{R}^{C\times T}$ is tokenized one channel at a time and attended to across channels. Unlike iTransformer, which treats variates as an interchangeable set sharing one embedding, ITX gives each channel a *dedicated, linear* embedding:

$$e^{(c)} = W_c\, x^{(c)} + b_c, \quad W_c \in \mathbb{R}^{d_{\text{model}}\times T}, \tag{3}$$

with W_c and b_c untied across c. The C tokens are processed by a standard Transformer encoder and decoded back to a window by per-channel linear heads:

$$H = \mathrm{Enc}\big(e^{(1)}, \ldots, e^{(C)}\big), \quad \hat{x}^{(c)} = W_c^{\text{out}} H^{(c)} + b_c^{\text{out}}. \tag{4}$$

Training by cross-instance channel substitution. The architecture alone does not enforce cross-channel inference. A plain reconstruction objective invites the identity shortcut: the model learns to copy its input rather than to infer each channel from the others, and the residual becomes uninformative. Corrupting the input penalizes this shortcut, but *how* it is corrupted matters: zeroing or adding noise, as in prior masked-reconstruction detectors (Ekambaram et al., 2026), leaves the corrupted channel trivially identifiable from its value alone, so the model only ever learns to *repair* a given position, not to find it. We instead corrupt a random subset of channels by *swapping* in values from other samples in the batch. Jeong et al. (2023) use a similar substitution to synthesize anomalies, but train the model to flag the replaced points; here the model is trained to recover the original values, so the corruption directly provides the repair target. Because each swapped-in value is a real sample from that channel's own marginal, it looks unremarkable in isolation. The model must first *detect* which channels are inconsistent with the rest of the array before it can *repair* them, which forces it to learn genuine cross-channel dependencies rather than a lookup keyed on which entries were corrupted.

For window x_i, a count $k \sim \mathrm{Unif}\{1, \ldots, K\}$ with $K = \lfloor C/3 \rfloor$ is drawn, and a mask $\mathbf{m} \in \{0,1\}^C$ marking k uniformly chosen channels is constructed. Resampling k per window varies how many sensors are corrupted at once, so the model never specialises to a single fault count. Corrupted channels are filled with values swapped in from other samples in the batch:

$$\tilde{x}_i^{(c)} = \begin{cases} x_i^{(c)}, & m_c = 0, \\ x_{j_c}^{(c)}, & m_c = 1, \end{cases} \tag{5}$$

where the donor index j_c is drawn per channel from an independent permutation of the batch indices $\{1, \ldots, B\}$.

The reconstruction $\hat{x}_i = h_\psi(\tilde{x}_i)$ is trained against the clean window:

$$\mathcal{L} = \frac{1}{BTC} \sum_{i=1}^{B} \sum_{t=1}^{T} \sum_{c=1}^{C} \big(\hat{x}_i(c,t) - x_i(c,t)\big)^2. \tag{6}$$

Scoring. The model never receives $\mathbf{m}$: corruption is applied to the raw input, so training and inference run the same forward pass. Scoring is $a(c,t) = \big(\hat{x}(c,t) - \tilde{x}(c,t)\big)^2$ with $\hat{x} = h_\psi(\tilde{x})$, one pass per window, reduced over time to the window-level $\bar{a}^{(c)}$ of Section 3.3.

5 RESULTS

5.1 CAN CURRENT DETECTORS ISOLATE SENSOR FAULTS?

Table 1 reports channel-level isolation averaged over the ten MuViS-C failure modes at severity 1.0. CATCH and CrossAD trade places depending on the dataset and metric. Neither dominates the other on precision, recall, or localization, and both are frequently matched or beaten by the simple 3σ threshold, most consistently on precision. ITX leads on recall for eight of the nine datasets, often by a wide margin over both the naive baselines and the ALoRa gates, and ALoRa-T is the only model to overtake it, on T66-14. Localization follows the same pattern. ITX is best on eight of nine datasets, with ALoRa-T again the exception, taking the lead on T66-14 and trailing ITX on Vehicle. The gap between ITX and the ALoRa variants is largest on Battery and MMR, where the ALoRa gates barely improve on, or fall below, the naive baselines.

Table 1: Channel-level precision, recall and localization (↑) at severity 1 and each channel's calibrated 1% FPR threshold, on the capped test sets (Section 3.1). Bold/underline: best/second-best.

Metric	Model	PM10	PM2.5	Batt.	HR	TEP	Veh.	MMR	T66-13	T66-14
Precision	3σ	.625	.625	**1.00**	.608	.490	.584	.430	.621	**.627**
	CATCH	.456	.456	**1.00**	.535	.402	.528	.331	.398	.454
	CrossAD	.476	.476	**1.00**	.623	.354	.607	.324	.445	.489
	ALoRa-T	.511	.511	.772	.623	.282	.883	.405	.502	.354
	ALoRa-Loc	.586	.586	.788	.680	.498	**.941**	.513	.572	.560
	ITX	**.726**	**.726**	.632	**.706**	**.535**	.882	**.537**	**.640**	.588
Recall	3σ	.299	.299	.559	.240	.523	.616	.296	.388	.408
	CATCH	.254	.254	.240	.350	.407	.429	.239	.248	.296
	CrossAD	.245	.245	.378	.213	.367	.499	.252	.294	.296
	ALoRa-T	.260	.260	.458	.277	.699	.752	.360	.348	**.700**
	ALoRa-Loc	.236	.236	.458	.264	.411	.733	.324	.320	.437
	ITX	**.634**	**.634**	**.958**	**.420**	**.807**	**.978**	**.768**	**.661**	.655
Localization	3σ	.527	.527	.779	.575	.604	.506	.465	.534	.502
	CATCH	.301	.301	.288	.445	.384	.560	.283	.282	.273
	CrossAD	.307	.307	.545	.490	.299	.507	.228	.245	.249
	ALoRa-T	.523	.523	.691	.573	.665	.889	.430	.487	**.675**
	ALoRa-Loc	.333	.333	.562	.413	.413	.792	.360	.360	.443
	ITX	**.758**	**.758**	**.944**	**.690**	**.743**	**.991**	**.766**	**.688**	.672

5.2 DOES ISOLATION TRANSLATE INTO DOWNSTREAM ROBUSTNESS?

Tables 2 and 3 first compare the six gates on the capped protocol, with ground-truth masking (Oracle) and ungated pass-through (Naive) as bounds, and then place ITX against the MuViS-C model pool on the full test sets. Among the gates, ITX ranks first on both mPC (1.4) and rPC (1.1) and recovers most of the available headroom, closing 80% of the oracle–naive gap on mPC (1.689 → 1.011 vs. 0.846) and 83% on rPC (2.081 → 1.218 vs. 1.041). 3σ ranks second, ahead of all other learned detectors.

Against the benchmark model pool, ITX achieves the best overall rank on both mPC (2.4) and rPC (2.2; significant over TST-ISensD, P-TSMixer on mPC; M-TCN, TST, TST-ISensD on rPC; Nemenyi, conservative at N=9), with rPC wins on five of nine datasets. On PM10, PM2.5, TEP and Vehicle it sits within 0.03 of the oracle, and on MMR it leads the pool with a larger margin to oracle (1.097 vs. 1.028). It trails on Batt., HR and the T66 pair (critical difference diagrams in Appendix O; per-failure-mode results in Appendix L).

5.3 GENERALIZATION TO A STANDARD ANOMALY BENCHMARK

Our results so far concern persistent faults on virtual-sensing data (Section 5.2). To check whether ITX's advantage transfers, we evaluate it on SMD (machine-1-1), a standard multivariate TSAD

Table 2: mPC (↓). Blocks use different test-set sizes (Section 3.1) and are ranked separately; GM is best-normalized. Oracle and Naive (gray) are unranked bounds.

Model	PM10	PM2.5	Batt.	HR	TEP	Veh.	MMR	T66-13	T66-14	Rank	GM
Gates — all six on the capped protocol											
3σ	106.25	70.00	0.06	5.36	**0.05**	5.37	0.18	0.12	0.13	3.2	1.306
CATCH	103.66	69.47	0.07	4.85	**0.05**	6.50	0.21	0.17	0.16	4.1	1.447
CrossAD	104.63	69.92	0.08	5.58	**0.05**	5.92	0.21	0.16	0.16	4.6	1.469
ALoRa-T	111.29	71.30	0.10	5.44	**0.05**	3.35	0.18	0.14	**0.10**	3.8	1.307
ALoRa-Loc	109.52	68.93	0.10	5.42	**0.05**	3.72	0.19	0.15	0.15	3.9	1.394
ITX (ours)	**101.35**	**67.15**	**0.02**	**4.78**	**0.05**	**3.26**	**0.14**	**0.11**	0.11	**1.4**	**1.011**
Oracle	96.04	63.51	0.01	3.93	0.05	3.11	0.12	0.09	0.09	–	0.846
Naive	114.13	69.55	0.13	6.10	0.05	7.32	0.22	0.19	0.19	–	1.689
MuViS-C baselines — full test-set resolution											
XGBoost	**97.12**	66.32	0.05	12.38	**0.05**	4.44	0.17	0.12	**0.11**	3.2	1.243
CatBoost	98.81	67.51	0.04	11.46	**0.05**	5.00	0.16	**0.11**	**0.11**	3.7	1.203
M-TCN	98.73	69.00	0.08	**5.27**	0.06	11.39	0.22	0.14	0.15	6.4	1.472
P-TSMixer	107.16	74.35	0.09	10.25	**0.05**	7.92	0.34	0.21	0.23	8.1	1.770
xLSTM-M	98.33	67.26	0.05	9.65	**0.05**	4.47	0.18	0.12	0.13	3.8	1.244
TST	106.23	73.51	0.06	12.52	**0.05**	8.17	0.21	0.14	0.16	6.8	1.508
TST-PGD	101.39	**65.98**	0.07	15.36	**0.05**	4.57	0.25	0.15	0.16	6.2	1.486
TST-ISensD	108.56	74.41	0.11	15.33	**0.05**	5.66	0.24	0.17	0.19	8.3	1.680
TST-F2F	118.44	68.75	**0.03**	23.53	0.07	4.96	0.18	0.12	0.12	6.1	1.383
ITX (ours), full test	101.35	67.15	**0.03**	5.30	**0.05**	**3.26**	**0.14**	**0.11**	**0.11**	**2.4**	**1.007**

Table 3: rPC (↓). Blocks as in Table 2; GM is a plain geometric mean.

Model	PM10	PM2.5	Batt.	HR	TEP	Veh.	MMR	T66-13	T66-14	Rank	GM
Gates — all six on the capped protocol											
3σ	1.062	1.075	7.038	1.628	1.008	1.797	1.472	1.372	1.422	2.9	1.598
CATCH	1.058	1.094	8.262	1.436	1.010	2.097	1.625	1.832	1.835	4.7	1.756
CrossAD	1.058	1.090	8.768	1.691	1.010	1.969	1.608	1.716	1.864	4.8	1.775
ALoRa-T	1.100	1.093	11.017	1.645	1.005	1.110	1.490	1.569	**1.104**	3.4	1.584
ALoRa-Loc	1.124	1.089	11.017	1.675	1.005	1.226	1.536	1.628	1.643	4.1	1.693
ITX (ours)	**1.025**	**1.042**	**2.454**	**1.435**	**0.999**	**1.042**	**1.097**	**1.145**	1.201	**1.1**	**1.218**
Oracle	1.017	1.017	1.052	1.216	0.997	1.031	1.028	1.015	1.010	–	1.041
Naive	1.209	1.114	14.489	1.889	1.016	2.423	1.823	2.041	2.166	–	2.081
MuViS-C baselines — full test-set resolution											
XGBoost	1.060	1.087	**2.159**	1.301	1.007	1.187	1.395	1.211	1.425	4.1	1.281
CatBoost	1.073	1.100	3.669	1.270	1.007	1.302	1.517	1.263	1.353	5.1	1.385
M-TCN	1.086	1.149	9.200	2.187	1.059	4.837	1.853	1.854	1.896	9.1	2.112
P-TSMixer	1.043	1.108	2.613	**1.099**	1.006	2.043	1.247	1.509	1.559	4.7	1.395
xLSTM-M	1.042	1.061	5.550	1.309	1.007	1.178	1.777	1.645	1.634	5.6	1.528
TST	1.091	1.155	3.761	1.817	1.010	2.157	1.897	1.858	1.948	8.9	1.716
TST-PGD	1.070	1.066	2.833	1.175	1.006	1.324	1.182	**1.108**	1.147	3.3	1.253
TST-ISensD	1.134	1.105	4.129	1.272	1.011	1.217	1.525	1.608	1.736	7.1	1.482
TST-F2F	1.095	1.102	2.208	1.521	1.231	1.159	1.202	1.126	**1.113**	4.9	1.272
ITX (ours), full test	**1.025**	**1.042**	2.181	1.460	**1.001**	**1.042**	**1.097**	1.145	1.201	**2.2**	**1.205**

benchmark unrelated to sensor faults, under the protocol of Shimillas et al. (2026) (Details in Appendix H). We change only the model, leaving their data, preprocessing, scoring, thresholding,

Table 4: SMD (machine-1-1) under ALoRa's protocol. Baselines copied from Tables 1–2, cost from Table 6 of Shimillas et al. (2026); ITX is scored by their code, ground truth and metrics. Best per column in bold, second best underlined. †Our timings were measured on different hardware.

	Detection			HR		NDCG		IPS		Cost	
Method	P	R	F1	@100	@150	@100	@150	@100	@150	Par. (M)	ms/sample
OmniAnomaly	0.68	0.66	0.67	0.29	0.46	0.24	0.34	0.17	0.26	–	–
Anom. Transf.	0.58	<u>0.88</u>	0.70	–	–	–	–	–	–	–	–
MEMTO	0.78	0.86	<u>0.79</u>	0.32	0.48	0.26	0.36	0.19	0.28	5.9	1.10
SARAD	0.88	0.67	0.78	0.44	0.56	0.47	0.55	<u>0.61</u>	0.74	9.6	0.11
ALoRa	**0.97**	**0.98**	**0.97**	**0.56**	**0.76**	**0.60**	**0.70**	0.60	<u>0.81</u>	3.2	0.13
ITX (ours)	<u>0.95</u>	**0.98**	**0.97**	<u>0.51</u>	<u>0.64</u>	<u>0.51</u>	<u>0.58</u>	**0.80**	**0.89**	0.47	0.01†

ground truth and metrics untouched, so their published numbers, reproduced in Table 4, are directly comparable to ours.

On detection, ITX and ALoRa both reach F1 0.97, clearly ahead of the next-best published method, MEMTO at 0.79. On localization, IPS credits a channel for its single best moment within an annotated event, matching the window-level isolation ITX is trained for, while HR and NDCG rank independently at every timestep, closer to conventional per-step TSAD scoring. ITX leads on IPS@100/150, reaching 0.80/0.89 against ALoRa's 0.60/0.81, and trails on HR/NDCG. It reaches this with raw reconstruction error. ITX is also smaller: 0.47M parameters, about $7\times$ fewer than ALoRa (3.2M) and 12–$20\times$ fewer than MEMTO and SARAD (5.9 and 9.6M). Its inference cost is roughly $10\times$ below the fastest baseline, though our timings (Apple M3 Max, MPS) and the published ones (likely $2\times$ V100) come from different hardware and are only indicative.

6 Limitations and Next Steps

Our approach relies on redundancy in the sensor array: if a faulty channel still carries information no other channel provides, discarding it is costly even when the detector is right. In future work we plan to quantify how much redundancy a given task offers, and determine when a faulty signal is better repaired or kept rather than dropped. We also focus on single-sensor faults, which lets us cleanly disaggregate the impact of each failure mode; extending the evaluation to simultaneous failures is a natural next step. Masking is also all-or-nothing: a flagged channel is dropped for the entire window, however briefly the fault occurs, so detecting faults per time step rather than per window is an extension that may particularly help with short anomalies like outliers. Finally, thresholds are calibrated once and then fixed, so they may drift out of calibration as operating conditions change; adapting thresholds online is another promising direction.

7 Conclusion

Sensor faults remain a serious threat to learning-based virtual sensing, as a single failing input can severely degrade the prediction. This work suggests that an effective defense is to recognize and discard such inputs rather than train the predictor to withstand them. Detecting and isolating the failing channel, then accommodating its absence with a predictor trained on simple input dropout, turns an open-ended robustness problem into two tractable ones. The results support this view. Paired with such a predictor, our proposed detector ITX makes explicit fault isolation the best-ranked robustification on the virtual sensing benchmark, and its localization ability carries over to a standard anomaly benchmark beyond sensor faults. More broadly, our findings show that how well a detector localizes is best judged by what its decisions do downstream. We therefore see end-to-end evaluation as a valuable complement to label-based metrics, and hope it will help steer time-series anomaly localization toward what deployed systems actually need.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 2623–2631, New York, NY, USA, July 2019. Association for Computing Machinery. ISBN 978-1-4503-6201-6. doi: 10.1145/3292500.3330701. URL https://dl.acm.org/doi/10.1145/3292500.3330701.

Edward Balaban, Abhinav Saxena, Prasun Bansal, Kai F. Goebel, and Simon Curran. Modeling, Detection, and Disambiguation of Sensor Faults for Aerospace Applications. *IEEE Sensors Journal*, 9(12):1907–1917, December 2009. ISSN 1558-1748. doi: 10.1109/JSEN.2009.2030284. URL https://ieeexplore.ieee.org/abstract/document/5297818. Conference Name: IEEE Sensors Journal.

Mogens Blanke, Michel Kinnaert, Jan Lunze, and Marcel Staroswiecki. *Diagnosis and Fault-Tolerant Control*. Springer Publishing Company, Incorporated, 3rd edition, July 2015. ISBN 978-3-662-47942-1.

Jens U. Brandt, Noah C. Pütz, Marcus Greiff, Thomas Jonathan Lew, John Subosits, Marc Hilbert, and Thomas Bartz-Beielstein. From Faults to Features: Pretraining to Learn Robust Representations against Sensor Failures. October 2025. URL https://openreview.net/forum?id=9aElHWiZ72.

Jens U. Brandt, Noah C. Puetz, Jobel Jose George, Niharika Vinay Kumar, Elena Raponi, Marc Hilbert, Thomas Bäck, and Thomas Bartz-Beielstein. MuViS: Multimodal Virtual Sensing Benchmark, March 2026a. URL http://arxiv.org/abs/2603.24602. arXiv:2603.24602 [eess].

Jens U. Brandt, Noah C. Puetz, Alexander Windmann, Marc Hilbert, Elena Raponi, Thomas Bäck, and Thomas Bartz-Beielstein. Reliable Virtual Sensing: A Multi-Domain Benchmark for Robustness Under Sensor Failures, September 2026b. URL http://arxiv.org/abs/2609.18396. arXiv:2609.18396 [cs.LG].

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15:1–15:58, July 2009. ISSN 0360-0300. doi: 10.1145/1541880.1541882. URL https://dl.acm.org/doi/10.1145/1541880.1541882.

Zhihao Dai, Ligang He, Shuanghua Yang, and Matthew Leeke. SARAD: Spatial Association-Aware Anomaly Detection and Diagnosis for Multivariate Time Series. November 2024. URL https://openreview.net/forum?id=gmf5Aj01Hz.

Janez Demšar. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006. ISSN 1533-7928. URL http://jmlr.org/papers/v7/demsar06a.html.

Luo Donghao and Wang Xue. ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis. October 2023. URL https://openreview.net/forum?id=vpJMJerXHU.

Vijay Ekambaram, Subodh Kumar, Arindam Jati, Sumanta Mukherjee, Tomoya Sakai, Pankaj Dayama, Wesley M. Gifford, and Jayant Kalagnanam. TSPulse: Tiny Pre-Trained Models with Disentangled Representations for Rapid Time-Series Analysis, March 2026. URL http://arxiv.org/abs/2505.13033. arXiv:2505.13033 [cs.LG].

Yiwei Fu and Weizhong Yan. One Masked Model is All You Need for Sensor Fault Detection, Isolation and Accommodation. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, June 2024. doi: 10.1109/IJCNN60899.2024.10649953. URL https://ieeexplore.ieee.org/document/10649953. ISSN: 2161-4407.

Robert Geirhos, Carlos R. Medina Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks, October 2020. URL http://arxiv.org/abs/1808.08750. arXiv:1808.08750 [cs.CV].

Mononito Goswami, Cristian Ignacio Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. Unsupervised Model Selection for Time Series Anomaly Detection. September 2022. URL https://openreview.net/forum?id=gOZ_pKANaPW.

Xiao Han, Saima Absar, Lu Zhang, and Shuhan Yuan. Root Cause Analysis of Anomalies in Multivariate Time Series through Granger Causal Discovery. October 2024. URL https://openreview.net/forum?id=k38Th3x4d9.

Rolf Isermann. *Fault-Diagnosis Systems*. Springer, Berlin, Heidelberg, 2006. ISBN 978-3-540-24112-6 978-3-540-30368-8. doi: 10.1007/3-540-30368-5. URL http://link.springer.com/10.1007/3-540-30368-5.

Yungi Jeong, Eunseok Yang, Jung Hyun Ryu, Imseong Park, and Myungjoo Kang. AnomalyBERT: Self-Supervised Transformer for Time Series Anomaly Detection using Data Degradation Scheme, May 2023. URL http://arxiv.org/abs/2305.04468. arXiv:2305.04468 [cs.LG].

Gonçalo Jesus, António Casimiro, and Anabela Oliveira. A Survey on Data Quality for Dependable Monitoring in Wireless Sensor Networks. *Sensors*, 17(9):2010, September 2017. ISSN 1424-8220. doi: 10.3390/s17092010. URL https://www.mdpi.com/1424-8220/17/9/2010. Number: 9.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002. ISSN 1046-8188. doi: 10.1145/582415.582418. URL https://dl.acm.org/doi/10.1145/582415.582418.

Tian Lan, Hao Duong Le, Jinbo Li, Wenjun He, Meng Wang, Chenghao Liu, and Chen Zhang. Towards Foundation Models for Zero-Shot Time Series Anomaly Detection: Leveraging Synthetic Data and Relative Context Discrepancy, May 2026. URL http://arxiv.org/abs/2509.21190. arXiv:2509.21190 [cs.LG].

Beibu Li, Qichao Shentu, Yang Shu, Hui Zhang, Ming Li, Ning Jin, Bin Yang, and Chenjuan Guo. CrossAD: Time Series Anomaly Detection with Cross-scale Associations and Cross-window Modeling, October 2025. URL http://arxiv.org/abs/2510.12489. arXiv:2510.12489 [cs.LG].

Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. Multivariate Time Series Anomaly Detection and Interpretation using Hierarchical Inter-Metric and Temporal Embedding. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, pp. 3220–3230, New York, NY, USA, August 2021. Association for Computing Machinery. ISBN 978-1-4503-8332-5. doi: 10.1145/3447548.3467075. URL https://dl.acm.org/doi/10.1145/3447548.3467075.

Qinghua Liu and John Paparrizos. The Elephant in the Room: Towards A Reliable Time-Series Anomaly Detection Benchmark. November 2024. URL https://openreview.net/forum?id=R6kJtWsTGy#discussion.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting, March 2024. URL http://arxiv.org/abs/2310.06625. arXiv:2310.06625 [cs].

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. February 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Dominik Martin, Niklas Kühl, and Gerhard Satzger. Virtual Sensors. *Business & Information Systems Engineering*, 63(3):315–323, June 2021. ISSN 1867-0202. doi: 10.1007/s12599-021-00689-w. URL https://doi.org/10.1007/s12599-021-00689-w.

Francisco Mena, Diego Arenas, and Andreas Dengel. Increasing the Robustness of Model Predictions to Missing Sensors in Earth Observation, September 2024. URL http://arxiv.org/abs/2407.15512. arXiv:2407.15512 [cs].

Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion Models for Adversarial Purification, May 2022. URL http://arxiv.org/abs/2205.07460. arXiv:2205.07460 [cs.LG].

Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. A Unifying Review of Deep and Shallow Anomaly Detection. *Proceedings of the IEEE*, 109(5):756–795, May 2021. ISSN 0018-9219, 1558-2256. doi: 10.1109/JPROC.2021.3052449. URL http://arxiv.org/abs/2009.11732. arXiv:2009.11732 [cs.LG].

Charalampos Shimillas, Kleanthis Malialis, Konstantinos Fokianos, and Marios M. Polycarpou. Low Rank Transformer for Multivariate Time Series Anomaly Detection and Localization, February 2026. URL http://arxiv.org/abs/2602.08467. arXiv:2602.08467 [cs.LG].

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 2828–2837, New York, NY, USA, July 2019. Association for Computing Machinery. ISBN 978-1-4503-6201-6. doi: 10.1145/3292500.3330672. URL https://dl.acm.org/doi/10.1145/3292500.3330672.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pp. 1096–1103, New York, NY, USA, July 2008. Association for Computing Machinery. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390294. URL https://dl.acm.org/doi/10.1145/1390156.1390294.

Xingjian Wu, Xiangfei Qiu, Zhengyu Li, Yihang Wang, Jilin Hu, Chenjuan Guo, Hui Xiong, and Bin Yang. CATCH: Channel-Aware Multivariate Time Series Anomaly Detection via Frequency Patching. October 2024. URL https://openreview.net/forum?id=m08aK3xxdJ.

A Reproducibility Statement

Our codebase, available at `https://anonymous.4open.science/r/LTI-52D4/README.md`, reproduces all main results in this paper via a single dependency-chain script (Appendix G). All nine datasets are MuViS-C's publicly released, open-source splits (Appendix D), and the ten failure modes used for corruption are MuViS-C's own unmodified implementations (Appendix E). Full hyperparameter search spaces and selection procedures are given in Appendix F and the final configurations are shipped with the code. Implementation notes and adaptation details for each baseline detector are as well provided in Appendix F. Downstream robustness metrics are computed by MuViS-C's own Testbed rather than a reimplementation. Experiments ran on standard hardware, Apple Silicon for development and short runs and NVIDIA H100 GPUs for full-scale runs. Compute cost and wall-clock estimates are reported in Appendix G. All random seeds are fixed per run and per hyperparameter trial.

B Large Language Model Usage

In this work, we used generative AI tools as a general-purpose writing and coding assistant. We additionally used generative AI tools for providing feedback on research methodology and experiments, implementing methods, and supporting qualitative and thematic data analysis. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, proposing or refining hypotheses, designing research methodology or experiments, assisting with translation, cleaning and reformatting datasets, or interpreting results, and generating synthetic datasets, formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools for creating or editing software code, creation of artifacts, drafting parts of the research paper, summarizing and analyzing existing literature, brainstorming, sourcing and searching for information, editing the research paper to improve readability, and identifying relevant literature. We have reviewed all AI-assisted work: we manually polished all AI-generated text, and we verified and tested all AI-generated code for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

C ITX Anomaly Localization Example

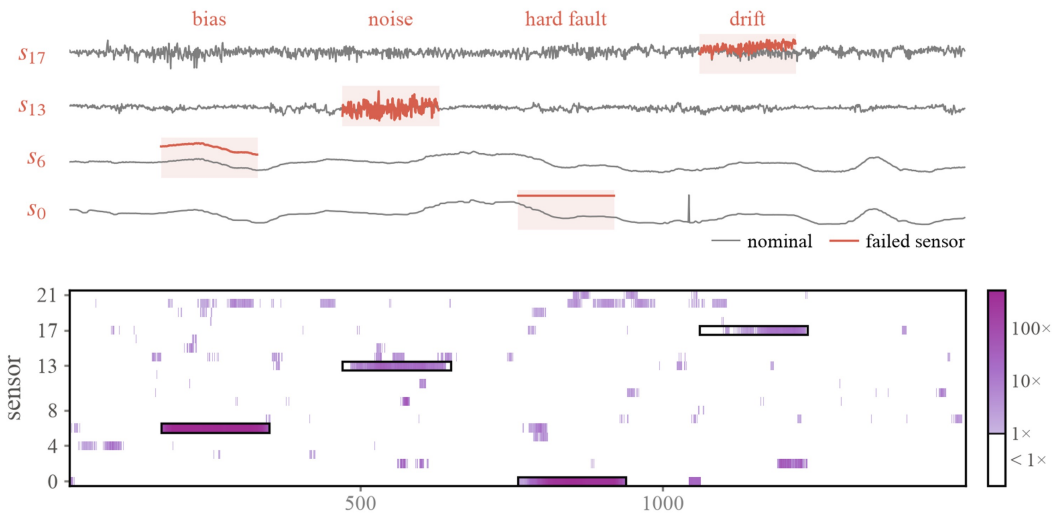


Figure 4: Sensor traces (top) and per-sensor score/threshold map (bottom) for ITX on a continuous REVS stream with four spliced-in failures, showing ground-truth boxes against detection that localizes correctly.

D DATA AND PREPROCESSING

We utilize the public **MuViS-C** (Brandt et al., 2026b) benchmark, and the datasets, the windowing, the train/test partition and the standardisation are all the benchmark's, used unmodified. MuViS-C casts each dataset as a *virtual-sensing* regression: given a length-T window of C input sensors, predict the value of a held-out scalar target at the end of the window (Beijing: PM10 / PM2.5 concentration; Panasonic: battery state of charge; PPG-DaLiA: heart rate; Tennessee Eastman: process variable `xmeas_35`; Vehicle Dynamics: lateral velocity `vyCG`; REVS: front-right tyre temperature). The target column is removed from the inputs, so the C in Table 5 counts input channels only.

Windowing. Raw series are cut into fixed-length windows by a stride-1 sliding window (stride 128 for PPG-DaLiA, whose windows are $T = 512$). Window length is a per-dataset property of the benchmark and is *not* constant: T ranges from 20 (Tennessee Eastman, REVS) to 512 (PPG-DaLiA).

Standardisation. A per-channel `StandardScaler` is fit on the training split (pooled over windows and timesteps) and applied to all three splits. Targets are left in their original units, matching MuViS-C, whose downstream learning rates were tuned against unscaled targets. All ten failure modes (Appendix E) assume z-scored inputs, which this step provides.

Table 5: Original benchmark datasets. C: features, T: sequence length, N: number of samples.

Dataset	**Domain**	C	T	N_{train}	N_{test}
BeijingPM10Quality	Air quality	9	24	11,918	5,048
BeijingPM25Quality	Air quality	9	24	11,918	5,048
Panasonic18650PFData	Battery SoC	7	120	199,827	158,126
PPGDalia	Wearable / Bio	6	512	51,757	12,940
REVS/2013 Monterey	Motorsport	22	20	120,777	21,357
REVS/2013 Targa 66	Motorsport	22	20	33,520	11,109
REVS/2014 Targa 66	Motorsport	22	20	46,739	9,036
TennesseeEastman	Chemical	33	20	240,500	470,500
VehicleDynamics	Automotive	11	50	1,384	280

E FAILURE CATALOGUE

The ten failure modes are MuViS-C's (Brandt et al., 2026b), imported and used unmodified. Severity $s \in [0, 1]$ is scaled internally by the benchmark's perturbation scale k, left at its published default k=3. We report at $s \in \{0.25,\ 0.5,\ 1.0\}$ for the evaluation of the anomaly detectors, and run the full twenty-step severity sweep for the virtual sensing robustness assessment. Each mode writes only to its single victim channel and leaves every other channel bit-identical. This is what lets the victim channel itself be the label, with no separate mask tensor.

Not every labelled positive is actually corrupted. The two trimming modes clip only the points that fall outside a band that narrows with severity, so at low severity a window can be assigned a victim channel and emerge byte-identical to the clean window. Asking a detector to distinguish a window from itself is not a meaningful test, so we compare the corrupted array against the clean one at a tolerance of $\epsilon = 10^{-4}$ and relabel untouched entries as negatives.

F IMPLEMENTATION DETAILS

F.1 SHARED TRAINING AND SELECTION PROTOCOL

Every baseline is used through its authors' released implementation, so the per-model subsections below document only what we added, changed, or had to choose. The four trained detectors (CATCH, CrossAD, ALoRa, ITX) additionally share one training harness and one model-selection protocol.

Harness. Each model trains for a fixed epoch budget E under a single half-cosine learning-rate decay $\eta_e = \frac{\eta_0}{2}\big(1+\cos(\pi e/E)\big)$, $e \in \{0, \ldots, E-1\}$ indexing completed epochs (CATCH additionally decays its mask-generator optimiser off its own base rate). Optimisation is Adam. The checkpoint

Table 6: The ten failure modes, as instantiated by `muvis_c.failures`. $t/(T-1) \in [0,1]$ indexes the window; $\mathcal{U}$ is uniform. Defaults: $k = 3$, $p_{\max} = 0.3$, $\alpha = 0.4$.

Mode	Effect on the victim channel
`bias`	constant offset $x' = x + sk$
`noise`	additive white Gaussian noise, $x' = x + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, (sk)^2)$
`scaling`	constant gain, $x' = x\,\big(1 + s(k-1)\big)$
`time_varying_scaling`	gain ramped over the window, $x' = x\,\big(1 + s(k-1)\,t/(T-1)\big)$
`linear_drift`	additive ramp $x' = x + sk\,t/(T-1)$, zero at the window start
`nonlinear_drift`	additive quadratic ramp $x' = x + sk\,\big(t/(T-1)\big)^2$, zero at the window start
`trimming_constant`	hard clip to the band $[-h, h]$, $h = (1-s)k$
`trimming_varying`	soft clip toward $[-h, h]$ with damping α (out-of-band values pulled $(1-\alpha)$ of the way in)
`outliers`	each timestep independently, with probability $s\,p_{\max}$, offset by $\mathcal{U}(-k, k)$
`hard_fault`	channel pinned to the constant sk (stuck-at)

kept is the lowest-validation-loss one. Seeds fix weight initialisation, batch shuffling and all injected randomness.

Model selection. Validation loss rewards reconstructing *everything* well, faults included, so it favours the identity-copy shortcut. We instead follow Goswami et al. (2022): score a configuration by how well it *detects* synthetically injected anomalies. The tuning faults (`flip`, `speedup`, `average`) are taken from that work and are disjoint from the ten evaluation modes. Selection is 50-trial Optuna TPE (Akiba et al., 2019) maximising pooled per-(window, channel) detection AUROC on those faults injected into the validation split, over four victim channels. Trials train on a 10% subsample of the training split; the winning configuration is then retrained on the full split by the harness above. The CATCH and CrossAD ranges are set around the settings in their own released configurations, while ITX and ALoRa are given a wider range of up to 200. Only 3 of the 36 selected configurations sit at their range maximum and 3 at their minimum, so the caps are largely not binding. No test data is touched at any point. The shared search space is in Table 7; model-specific additions are listed per model, and the final per-dataset configurations are released with the code.

Table 7: Shared Optuna search space. Braces: categorical sets; brackets: integer or continuous ranges; learning rates are log-uniform. d_{model} is drawn from the powers of two in the stated range. Patch sizes are $\lfloor T/k \rfloor$ for integer k, so a window holds nominally the stated number of tokens. Model-specific dimensions and pinned choices are given in the per-model subsections.

	CATCH	**CrossAD**	**ITX**	**ALoRa**
learning rate	$[10^{-4},\ 3{\cdot}10^{-3}]$	$[10^{-5},\ 10^{-3}]$	$[5{\cdot}10^{-5},\ 10^{-3}]$	$[5{\cdot}10^{-5},\ 10^{-3}]$
epochs	$[5, 20]$	$[5, 25]$	$[40, 200]$	$[20, 200]$
d_{model}	16–128	32–256	64–512	fixed†
$d_{\text{ff}}/d_{\text{model}}$	$\{1, 2, 4\}$	4	$\{2, 4\}$	no FFN
enc. / dec. layers	$[1, 4]$ / –	$[1, 3]$ / $[1, 3]$	$[1, 4]$ / –	$[1, 4]$ / –
heads	$\{2, 4, 8\}$	$\{2, 4, 8\}$	$\{4, 8, 16\}$	$\{2, 4, 8\}$
dropout	$[0, 0.3]$	$[0, 0.3]$	$[0, 0.3]$	$[0, 0.3]$
tokens per window	4–16	4–20	–	–
batch size (fixed)	256	256	1024	1024

† Fixed by the pairwise embedding to $\min\big(C(C-1)/2,\ 512\big)$.

F.2 CATCH

We use the authors' released architecture implementation unchanged. Beyond the shared space, the search covers the patch stride (non-overlapping or 50% overlap) and three loss weights: the frequency-domain auxiliary term, the weight of that term in the anomaly score, and the internal

balance of the cross-channel contrastive loss; the outer weight on that contrastive loss stays at the authors' value. The anomaly score is the per-channel temporal error plus the frequency-domain error, averaged over four forward passes. Selected architectures span $d_{\text{model}} \in [16, 128]$, 1–4 encoder layers, 2–4 heads and patch sizes of 2–85 across the nine datasets. Two deviations from the reference training loop: (i) the half-cosine schedule replaces the per-epoch exponential halving; (ii) the mask-generator optimiser is stepped *after* `loss.backward()`, accumulated over $\approx 10\%$ of an epoch, an MPS correctness fix.

F.3 CROSSAD

We use the authors' released architecture implementation unchanged. One of its properties is load-bearing for our results: CrossAD is *channel-independent by construction*, since its forward pass reshapes $(\text{batch}, \text{channels})$ together and therefore carries no cross-channel information at all. Beyond the shared space, the search covers the multi-scale pooling kernels (fine / medium / coarse sets, sampled relative to T), the query top-$k \in [1, 5]$ and the query-bank size $\in \{8, 16, 32\}$. Selected architectures span $d_{\text{model}} \in [32, 256]$, 1–3 encoder and 1–3 decoder layers, 2–8 heads and patch length 2–42. The only training-loop changes is the switch to the half-cosine schedule.

F.4 THE 3σ REFERENCE

The Shewhart control chart scores each element by its distance from the channel's healthy mean in units of that channel's healthy standard deviation, and flags the channel if any sample in the window leaves the control limits:

$$a(c,t) = |\tilde{x}(c,t) - \mu_c|/\sigma_c, \qquad d_c = 0 \iff \max_t a(c,t) > \tau_c,$$

with μ_c, σ_c a closed-form fit on the clean training split, pooled over windows and timesteps. It models each sensor entirely on its own, which is the point: it is the cleanest available test of whether cross-channel machinery earns its keep. Its window aggregation is $\max_t$, in place of the time average of Equation 1. We calibrate τ_c per channel at the shared 1% operating point.

F.5 ITX

The encoder of Equation 4 is a standard pre-norm Transformer encoder, and there is no positional encoding: the token axis is the sensor array, which carries no order to encode. The model was trained with the shared harness at a tuned budget of 41–194 epochs, with a fixed-seed mask and fixed-seed donors at validation so that epochs are comparable. Selected architectures span $d_{\text{model}} \in \{64, 128, 512\}$, 1–4 layers and $\{4, 8\}$ heads. The corruption strategy, the loss target and K are pinned.

F.6 ALORA

We import the authors' released architecture unchanged, and the training loss and the two scoring rules follow their released codebase. Their pairwise LightMTS-Embed makes each token a channel pair, so $d_{\text{model}} = \min\big(C(C-1)/2,\ 512\big)$, rounded down to even, and only Tennessee Eastman (528 pairs) hits the cap. Depth, heads, learning rate and λ_{reg} are searched rather than taken from their published setup, which is fixed across the datasets they report and was tuned at d_{model} 512–702, far above our regime (selected: 1–4 layers, 2–8 heads, 20–195 epochs, λ_{reg} between $6{\cdot}10^{-8}$ and $9{\cdot}10^{-3}$). The low-rank penalty is truncated at $r{=}1$, the Geman constant γ is searched alongside. The indicator threshold h_1 is calibrated after training on clean training windows as the maximum of the fifth-largest singular value, as they prescribe.

Both scoring rules are adapted to our per-channel unit. ALoRa-T keeps the squared reconstruction error per channel rather than summing over channels, which leaves the attention-rank factor a per-window scalar and therefore unable to change the ranking across channels within a window. Our ALoRa-Loc implementation is a two-stage cascade: stage 1 flags channels that clear the shared 1% threshold, and stage 2 applies their localization procedure to keep a single channel among them. This is one way to operationalize a per-dimension attribution as a channel-level decision, and on this benchmark it favours the method, since the cascade emits exactly one channel per window and our

battery perturbs exactly one. With several sensors failing at once, a top-k rule would need k, which a detector does not know.

F.7 MODERNTCN

The downstream regressor behind the ignore-gate is MuViS-C's own ModernTCN, with the benchmark's own per-dataset tuned architecture and learning rate. Across the nine datasets their configurations use 2–4 residual blocks of grouped depthwise convolutions at $d_{\text{model}} \in \{32, 64, 128\}$, with large kernels 7–71, small kernels 3–25, FFN ratio 1–4, no patching, and a linear head to the scalar target. Training is Adam at their tuned per-dataset rate ($3{\cdot}10^{-5}$ to $8{\cdot}10^{-3}$), MSE loss, 100 epochs, half-cosine decay, batch 256, and the best checkpoint by *clean* validation loss. The only training-time input perturbation is that ISENSD input-sensor dropout: each channel is zeroed with probability 0.3, with a guard that force-keeps one channel in any window that lost all of them. That rate is our one departure from their configurations, which use the equivalent of 0.5.

G COMPUTE AND REPRODUCTION

Hardware and cost. All experiments run on NVIDIA H100 GPUs, except those on SMD, which run on a MacBook Pro (M3 Max, 128 GB). Device selection resolves CUDA, then Apple MPS, then CPU. The full pipeline, detector and sensor training plus both evaluation sweeps, all nine datasets, takes roughly six days of wall-clock time at three concurrent GPUs, dominated by the Testbed sweeps on Panasonic, PPG-DaLiA and Tennessee Eastman. A single small dataset end-to-end (Vehicle Dynamics) takes under 20 minutes, which is the recommended check before committing cluster time.

Where the numbers are a subsample. Three datasets are evaluated on a subset of their test split; the same subset is used for *every* gate, so each comparison is between methods and not between test sets. The remaining six use their full test split.

Dataset	Eval.	Avail.	Binding constraint (uncapped)
PPG-DaLiA	2,048	12,940	ALoRa's per-batch SVD: $\sim$17 d
Panasonic	12,288	158,126	as above: $\sim$34 d
Tennessee Eastman	40,000	470,500	C=33 victims: $\sim$2.5 d per model, $\sim$20 d for one sweep

Reproduction. The code exposes five entry points, detector training, tuning and evaluation, sensor training, and end-to-end gated evaluation, plus one script that submits the whole dependency chain and writes every main table and figure in this paper in one results folder. The table scripts read the per-dataset result CSVs directly, so tables and figures rebuild in seconds without re-running a model. Hyperparameters are committed, nothing is searched at reproduction time, and all reported numbers use seed 42.

H SMD DETAILS

We restate the protocol of Shimillas et al. (2026) as we run it, since every number we report and every baseline we quote depends on it.

Data. SMD here means machine-1-1 alone: 28,479 train and 28,479 test rows over $d = 38$ sensors. A single `StandardScaler` is fit on train and applied to test, and both splits are cut into stride-1 windows of length 20.

Score assembly. Our model emits a per-timestep, per-channel squared reconstruction error. Overlapping windows are collapsed by taking all 20 timesteps of the first window and only the last timestep of every subsequent one, giving one row per test timestep. Detection averages this matrix over channels; localization keeps it.

Detection. Precision and recall are affiliation-based. The threshold is the best of 200 candidates at score percentiles 70 to 100, chosen by F1 against the test labels.

Upper bounds. The detection threshold is selected on the test labels and the ranking metrics are given the ground truth's own $|G_t|$. This is their stated convention and we apply it unchanged, so every row of Table 4, theirs and ours alike, is an upper bound rather than an achievable operating point.

Localization Metrics We use the three localization metrics of Shimillas et al. (2026). Each one ranks the d channels by a per-channel anomaly score $s_t^{(i)}$ and compares the top of that ranking with $G_t \subseteq \{1, \dots, d\}$, the set of channels labelled anomalous at time t. A tolerance $P \in \{100, 150\}$ sets how far down the ranking is read. Let Γ_t^P be the $k_t = \lceil |G_t| \cdot P/100 \rceil$ highest-scoring channels. At $P = 100$ the method gets exactly as many guesses as there are anomalous channels, and at $P = 150$ it gets half as many again.

Hit Rate (HR@P). HR (Su et al., 2019) is the fraction of anomalous channels that appear in the top k_t:

$$\mathrm{HR}_t@P = \frac{|G_t \cap \Gamma_t^P|}{|G_t|}. \tag{7}$$

NDCG@P (Järvelin & Kekäläinen, 2002) also accounts for rank order. Let $r_j \in \{0, 1\}$ indicate whether the j-th ranked channel in Γ_t^P is in G_t. Then

$$\mathrm{DCG}_t@P = \sum_{j=1}^{k} \frac{r_j}{\log_2(j+1)}, \qquad \mathrm{IDCG}_t = \sum_{j=1}^{|G_t|} \frac{1}{\log_2(j+1)}, \tag{8}$$

$$\mathrm{NDCG}_t@P = \frac{\mathrm{DCG}_t@P}{\mathrm{IDCG}_t} \in [0, 1]. \tag{9}$$

Interpretation Score (IPS@P) (Li et al., 2021) evaluates localization at the segment level. For each anomalous segment S_i, every channel j is scored by $s_{S_i}^{(j)} = \max_{t \in S_i} s_t^{(j)}$. The top-ranked channels P_{S_i} are then compared with the ground-truth channels G_{S_i}:

$$\mathrm{IPS}@P = \frac{1}{N} \sum_{i=1}^{N} \frac{|G_{S_i} \cap P_{S_i}|}{|G_{S_i}|}, \tag{10}$$

where N is the number of anomalous segments, each weighted equally.

I ITX COMPONENT ABLATIONS

Table 8 adds ITX's four departures from a tied-weight inverted-attention autoencoder one at a time; each rung trains at a single seed from ITX's tuned configuration (shared encoder, full schedule), and precision/recall/rPC come from a subsampled Testbed sweep (1024 windows, 5 severities), internal to the ablation and not comparable to Table 3. Removing RevIN helps only on Vehicle, where precision and recall both jump sharply, and is roughly flat elsewhere. Cross-instance substitution lifts Beijing precision and recall over zero-masking; per-channel embeddings then give the largest single gain, raising recall on all three datasets, and untying the per-channel output projections on top of that further raises precision and recall everywhere. Row 6 (ITX) is best on nearly every metric across the three datasets, the one exception being REVS precision, where the zero-masking variant (row 3) scores higher but at far lower recall. Downstream, rPC tracks detection where the gate matters (Vehicle $2.07 \to 1.04$, REVS $1.76 \to 1.12$) and is flat on Beijing, already near-oracle ungated.

	Configuration				Beijing			Vehicle			REVS		
#	RevIN	corr.	p.s. emb.	p.s. proj.	Prec.	Rec.	rPC	Prec.	Rec.	rPC	Prec.	Rec.	rPC
1	✓	–			0.521	0.300	1.046	0.466	0.372	2.069	0.539	0.198	1.760
2		–			0.528	0.291	1.048	0.786	0.792	1.213	0.550	0.211	1.636
3		zero			0.467	0.291	1.050	0.780	0.569	1.564	**0.735**	0.204	1.652
4		subst.			0.670	0.359	1.049	0.773	0.778	1.144	0.523	0.295	1.458
5		subst.	✓		0.688	0.560	**1.031**	0.751	0.868	1.057	0.547	0.538	1.262
6		subst.	✓	✓	**0.749**	**0.610**	**1.031**	**0.894**	**0.975**	**1.038**	0.646	**0.638**	**1.121**
naive / *oracle* rPC					1.200 / 1.015			2.325 / 1.004			1.970 / 1.002		

Table 8: Ablation ladder from a plain inverted-attention autoencoder (row 1) to ITX (row 6). *zero*/*subst.*: alternative training corruptions; *p.s. emb./proj.*: per-sensor instead of shared embedding / output projection. Precision and recall at severity 1, rPC from a reduced sweep, both on the first 1,024 test windows (Vehicle: all 280); single seed.

J SUBSET ABLATION

ALoRa's per-batch SVD puts a full Testbed sweep at roughly 17 (HR) and 34 (Battery) days, so the models are evaluated on a subset of these test splits. We additionally report the full test set results of the other models here for the sake of completeness. Absolute values shift between the full split and the prefix (Tables 9 and 10), but the position of ITX does not. It ranks first on recall, localization, mPC and rPC on both datasets under both arms. ITX does not lead in Battery precision, in both arms.

Table 9: Channel-level detection at severity 1 on the full test sets (Full) and on the capped ones. Same checkpoints, seed and calibration in both arms; only the evaluated windows differ.

		3σ		**CATCH**		**CrossAD**		**itx**	
Dataset	**Metric**	Full	Sub.	Full	Sub.	Full	Sub.	Full	Sub.
HR	Precision	0.579	0.608	0.556	0.535	0.566	0.623	0.712	0.706
	Recall	0.246	0.240	0.355	0.350	0.218	0.213	0.418	0.420
	Localization	0.545	0.575	0.427	0.445	0.451	0.490	0.675	0.690
Batt.	Precision	0.786	1.000	0.597	1.000	0.768	1.000	0.633	0.632
	Recall	0.454	0.559	0.250	0.240	0.310	0.378	0.905	0.958
	Localization	0.632	0.779	0.300	0.288	0.397	0.545	0.908	0.944

K DOES TRAINING-TIME MASKING STILL MATTER BEHIND A GATE?

ISENSD's training-time channel dropout forces the virtual sensor to learn from windows where some channels are randomly zeroed, on the argument that this teaches it to tolerate the channels an inference-time gate will later zero for real. But once a gate is already doing that zeroing, does the training-time masking still pay for itself, or has the gate made it redundant? We isolate this by training a second sensor identical in every other respect (architecture, weight init, data shuffling) but with `sensd_ratio`$= 0$, i.e. no training-time masking at all, and evaluating both sensors under the same three conditions: ungated (*Naive*), gated by ITX, and gated by ground truth (*Oracle*), against the identical corrupted arrays. Table 11 (mPC) shows a clear split. Behind either gate, masking during training is a consistent win. Ungated, the pattern reverses: with no gate to make the zeroed-channel skill relevant, the masking-trained sensor's mPC is on average higher than the non-masked one. Training-time masking, in other words, is not a general-purpose robustness trick, it specifically teaches the sensor to make use of what the gate leaves behind, and that skill only pays off once a gate is actually there to invoke it.

Table 10: Downstream results behind the same gates.

Model	Test set	HR			Batt.		
		Nominal	mPC	rPC	Nominal	mPC	rPC
3σ	Full	3.607	6.071	1.683	0.01110	0.0810	7.301
	Sub.	3.294	5.360	1.628	0.00865	0.0609	7.038
CATCH	Full	3.611	5.453	1.510	0.01108	0.0756	6.819
	Sub.	3.378	4.851	1.436	0.00865	0.0715	8.262
CrossAD	Full	3.591	6.292	1.752	0.01109	0.0869	7.834
	Sub.	3.301	5.583	1.691	0.00865	0.0758	8.768
ITX	Full	3.628	5.297	1.460	0.01221	0.0266	2.181
	Sub.	3.333	4.782	1.435	0.00865	0.0212	2.454
Naive	Full	3.526	6.783	1.924	0.01108	0.1412	12.747
	Sub.	3.230	6.103	1.889	0.00865	0.1253	14.489
Oracle	Full	3.526	4.259	1.208	0.01108	0.0123	1.111
	Sub.	3.230	3.929	1.216	0.00865	0.0091	1.052

Table 11: mPC, absolute (lower is better). Naive/ITX/Oracle each shown for both sensors: the production ISENSD-trained modernTCN (Mask) and the same architecture trained with no input-sensor dropout (No mask).

Dataset	Naive		ITX		Oracle	
	Mask	No mask	Mask	No mask	Mask	No mask
PM10	114.1	98.58	101.4	107.1	96.04	97.9
PM2.5	69.55	69.18	67.15	78.79	63.51	70.83
Batt.	0.1253	0.06289	0.02123	0.05003	0.0091	0.04207
HR	6.103	5.498	4.782	5.797	3.929	6.735
TEP	0.05336	0.05591	0.05253	0.05678	0.05239	0.05518
Veh.	7.316	14.01	3.257	12.32	3.114	11.9
MMR	0.2185	0.2127	0.1384	0.3531	0.1232	0.1841
T66-13	0.1874	0.1444	0.1061	0.1339	0.09319	0.1147
T66-14	0.1941	0.156	0.1086	0.1731	0.09048	0.1253

L VIRTUAL SENSING RESULTS PER FAILURE MODE

Table 12 breaks the headline rPC down by failure mode. ITX is the best gate on eight of ten modes, with the largest margins on sustained faults (`hard_fault`, `linear_drift`, `nonlinear_drift`, `bias`: 0.33–0.50 below the runner-up). CATCH wins `noise` and `outliers`, with ITX a close second ($+0.03$ and $+0.002$), but on `bias` and `hard_fault` CATCH barely improves on Naive. On trimming, all baselines match Naive, while ITX closes most of the gap to the Oracle, which remains best on every mode.

M NOMINAL PERFORMANCE

Table 13 reports RMSE on clean test data. There, any mask is a false alarm, and Oracle coincides with Naive, since both reduce to the ISENSD back-end on the full array; we therefore report Naive only. Because every row shares this back-end, the spread between gates isolates the cost of false alarms, and it is small: all six gates reach a GM between 1.004 and 1.025, against 1.002 for Naive. ITX ranks last among the gates on clean data (GM 1.025), consistent with its higher recall under corruption coming at a slightly higher false-alarm cost, but its intervals overlap those of Naive on every dataset, on MMR and Batt. only at the boundary.

Table 12: rPC (↓) per failure mode on the capped test sets, geometric mean over the nine datasets. Oracle and Naive (gray) are references.

	Gates						Reference	
Failure mode	**3σ**	**CATCH**	**CrossAD**	**ALoRa-T**	**ALoRa-Loc**	**ITX (ours)**	Oracle	Naive
bias	2.073	2.747	2.567	1.689	1.886	**1.362**	1.044	2.760
hard fault	2.239	2.493	2.484	1.696	1.908	**1.195**	1.044	2.495
linear drift	1.738	1.939	1.955	1.677	1.790	**1.200**	1.044	2.098
nonlinear drift	1.575	1.728	1.772	1.605	1.687	**1.184**	1.044	1.868
noise	1.502	**1.091**	1.368	1.991	2.200	1.120	1.044	2.868
outliers	1.464	**1.136**	1.485	1.589	1.603	1.138	1.043	1.623
scaling	1.473	1.980	1.755	1.489	1.625	**1.385**	1.044	2.231
time-var. scaling	1.349	1.553	1.524	1.420	1.496	**1.259**	1.044	1.751
trimming (const.)	1.057	1.055	1.056	1.056	1.057	**1.030**	1.027	1.056
trimming (varying)	1.057	1.056	1.057	1.056	1.058	**1.035**	1.027	1.058

Table 13: Nominal RMSE (↓) on clean test windows (capped test sets), bootstrap mean with 95% CI in gray (n_{boot}=200). All six run the same ModernTCN sensor, so the spread between rows is the price of gating, and the whole table is one ranked pool: Rank is autorank's mean rank over the six, GM the geometric mean of value ÷ the per-column best.

Model	**PM10**	**PM2.5**	**Batt.**	**HR**	**TEP**	**Veh.**	**MMR**	**T66-13**	**T66-14**	**Rank**	**GM**
	106.492	71.468	0.009	3.460	0.053	3.152	0.127	0.093	0.089		
3σ (Shewhart)	100.082	65.095	**0.009**	3.294	0.053	**2.987**	0.125	**0.090**	**0.088**	3.1	1.014
	93.408	58.938	0.009	3.131	0.052	2.815	0.122	0.088	0.086		
	104.813	70.089	0.009	3.548	0.053	3.278	0.130	0.094	0.091		
CATCH	97.933	63.525	**0.009**	3.378	**0.052**	3.099	0.128	0.092	0.089	3.8	1.020
	92.180	58.155	0.009	3.224	0.052	2.936	0.125	0.089	0.088		
	105.700	70.366	0.009	3.471	0.053	3.171	0.134	0.095	0.090		
CrossAD	98.896	64.169	**0.009**	3.301	**0.052**	3.008	0.131	0.092	**0.088**	3.2	1.017
	92.477	58.272	0.009	3.156	0.052	2.846	0.128	0.089	0.087		
	107.433	71.619	0.009	3.471	0.053	3.181	0.124	0.094	0.091		
ALoRa-T	101.144	65.213	**0.009**	3.306	**0.052**	3.019	0.122	0.091	0.089	3.7	1.014
	94.590	59.422	0.009	3.151	0.052	2.853	0.119	0.088	0.088		
	103.576	69.431	0.009	3.399	0.053	3.189	0.124	0.093	0.090		
ALoRa-Loc	**97.469**	**63.274**	**0.009**	**3.240**	**0.052**	3.033	**0.121**	0.091	0.089	**2.3**	**1.004**
	91.626	57.802	0.009	3.087	0.052	2.869	0.119	0.088	0.088		
	106.281	70.606	0.009	3.508	0.053	3.314	0.132	0.095	0.092		
ITX (ours)	98.906	64.451	**0.009**	3.333	0.053	3.127	0.126	0.093	0.090	4.9	1.025
	92.626	58.877	0.009	3.181	0.052	2.960	0.122	0.089	0.089		
	100.966	68.484	0.009	3.381	0.053	3.181	0.122	0.095	0.091		
Naive	94.422	62.420	0.009	3.230	0.053	3.019	0.120	0.092	0.090	–	1.002
	88.482	57.429	0.009	3.069	0.052	2.858	0.118	0.089	0.088		

N Anomaly Localization at different severities

Tables 14 and 15 repeat Table 1 at severities 0.25 and 0.5. At s=0.25, a bias amounts to only 0.75 training standard deviations, well inside the nominal range. All detectors lose recall and localization as severity drops, but ITX degrades most gracefully. At both severities, it remains best on localization on eight of nine datasets and on recall on seven, and its margin widens. At s=0.25, its Battery recall is .569 against .182 for the next-best gate.

Table 14: As Table 1, at severity 0.25.

Metric	Model	PM10	PM2.5	Batt.	HR	TEP	Veh.	MMR	T66-13	T66-14
Precision	3σ	.340	.340	**1.00**	.363	.165	.407	.146	.252	.250
	CATCH	.296	.296	**1.00**	**.543**	.179	.302	.143	.186	.237
	CrossAD	.301	.301	.600	.217	.200	.344	.186	.240	.208
	ALoRa-T	.185	.185	.685	.288	.261	.632	.085	.115	.343
	ALoRa-Loc	.213	.213	.723	.337	**.314**	.691	.110	.135	**.419**
	ITX	**.404**	**.404**	.568	.278	.286	**.762**	**.334**	**.289**	.258
Recall	3σ	.085	.085	.182	.057	.079	.278	.057	.063	.066
	CATCH	.139	.139	.169	**.323**	.104	.201	.063	.070	.085
	CrossAD	.107	.107	.052	.011	.131	.182	.111	.097	.091
	ALoRa-T	.027	.027	.016	.023	.237	.253	.033	.028	**.287**
	ALoRa-Loc	.021	.021	.015	.020	.151	.245	.023	.021	.221
	ITX	**.185**	**.185**	**.569**	.040	**.294**	**.610**	**.334**	**.169**	.175
Localization	3σ	.293	.293	**.667**	.327	.141	.140	.114	.155	.136
	CATCH	.312	.312	.469	.419	.132	.360	.104	.108	.113
	CrossAD	.300	.300	.418	.286	.117	.223	.109	.111	.118
	ALoRa-T	.211	.211	.277	.225	.189	.600	.083	.099	**.263**
	ALoRa-Loc	.112	.112	.130	.203	.163	.316	.056	.055	.233
	ITX	**.426**	**.426**	.592	**.446**	**.211**	**.787**	**.345**	**.245**	.223

Table 15: As Table 1, at severity 0.5.

Metric	Model	PM10	PM2.5	Batt.	HR	TEP	Veh.	MMR	T66-13	T66-14
Precision	3σ	.460	.460	**1.00**	.484	.337	.471	.245	.403	.396
	CATCH	.364	.364	**1.00**	**.575**	.288	.396	.219	.284	.327
	CrossAD	.364	.364	.889	.335	.279	.449	.241	.317	.312
	ALoRa-T	.321	.321	.705	.468	.283	.747	.179	.269	.357
	ALoRa-Loc	.370	.370	.721	.541	.388	.773	.238	.322	**.488**
	ITX	**.570**	**.570**	.579	.464	**.441**	**.843**	**.442**	**.471**	.430
Recall	3σ	.145	.145	.318	.120	.250	.411	.122	.154	.155
	CATCH	.188	.188	.230	**.334**	.251	.259	.122	.131	.144
	CrossAD	.171	.171	.159	.042	.264	.326	.154	.152	.150
	ALoRa-T	.072	.072	.112	.066	.431	.522	.078	.080	**.473**
	ALoRa-Loc	.062	.062	.112	.062	.258	.512	.063	.070	.331
	ITX	**.386**	**.386**	**.735**	.138	**.532**	**.797**	**.546**	**.370**	.374
Localization	3σ	.325	.325	.662	.409	.326	.275	.225	.278	.241
	CATCH	.300	.300	.342	.441	.257	.488	.173	.181	.181
	CrossAD	.289	.289	.339	.357	.226	.339	.170	.177	.187
	ALoRa-T	.336	.336	.516	.402	.410	.763	.179	.234	**.464**
	ALoRa-Loc	.152	.152	.250	.234	.265	.601	.095	.106	.337
	ITX	**.591**	**.591**	**.775**	**.559**	**.441**	**.887**	**.543**	**.424**	.403

O Statistical Significance

We test mean-rank differences with a Friedman test over N=9 datasets and k=10 systems (Oracle and Naive excluded), followed by a Nemenyi post-hoc test at α=0.05 (Figure 5 and Figure 6).

P Severity Curves

Figure 7 shows, per dataset, the test RMSE under corruption along the severity axis, averaged over failure modes and victim channels. It traces how prediction error grows as faults become stronger. On several datasets, the error of gated sensors first rises and then falls again at high severity, presumably because faults become large enough for the detectors to flag them reliably.

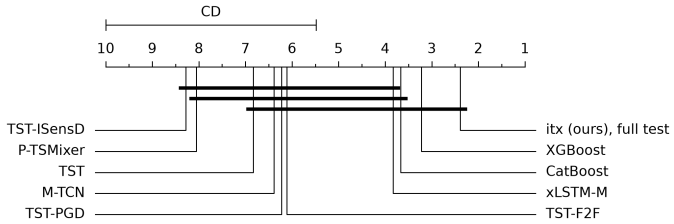


Figure 5: Critical difference diagram for ITX and benchmark baselines. mPC.

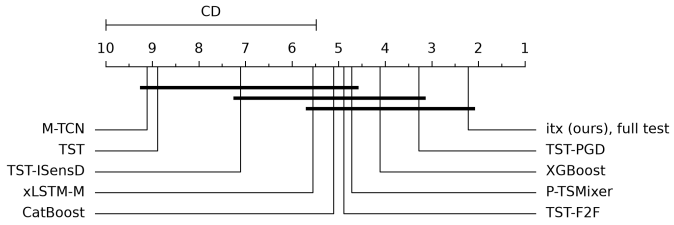


Figure 6: Critical difference diagram for ITX and benchmark baselines. rPC.

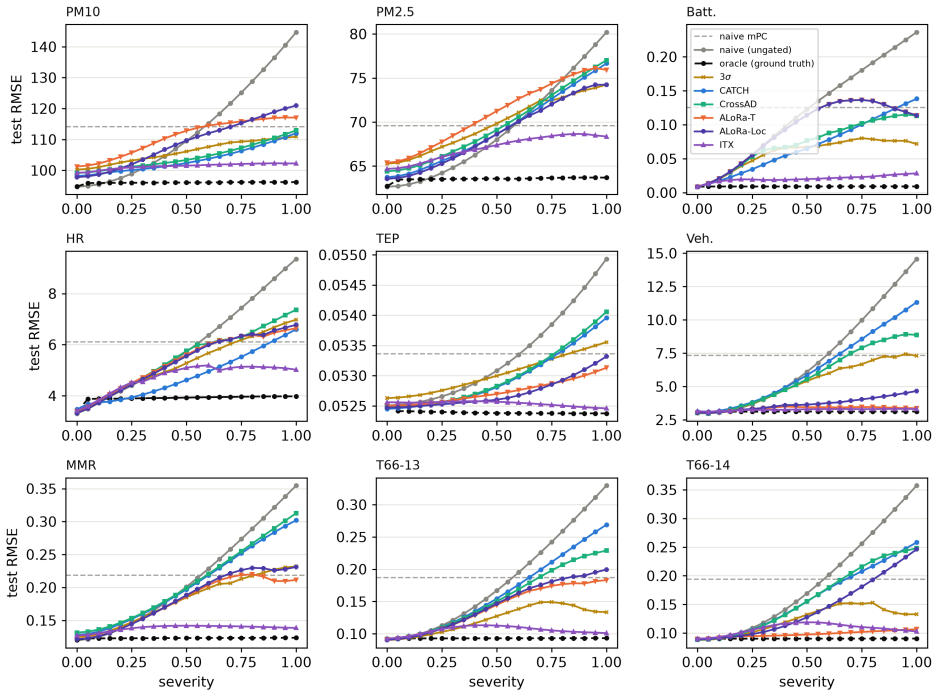


Figure 7: Severity curves of all gated virtual sensors. One cell per dataset.