FORECAST OR RECALL? ATTRIBUTING TRAINING DATA MEMORIZATION IN TIME SERIES FOUNDATION MODELS WITH TIMECANARY

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series foundation models are evaluated on public benchmarks, yet their pre-training data are seldom documented at the series level. Low forecast error can thus reflect generalization or recall, leaving data owners unable to tell whether a model learned from their series. We find that a post-training audit depends on the baseline forecaster and lacks enough valid comparison pairs for a reliable conclusion. We introduce **TimeCanary**, which marks an owner's original series before release with a sparse Fourier signature scaled to each series. The mark extends across training windows. For each marked span, a recorded coin flip chooses one of two candidate keys for the mark; the other key provides a comparison. We score both keyed windows by model loss and use a matched model trained with reversed key roles to cancel pre-existing key differences. The test also requires reference models and diagnostic checks. To our knowledge, this is the first effective method shown to attribute use of marked original time series in a controlled audit. The paired score met its support rule across diverse models, including when only one quarter of eligible host spans were marked.

1 INTRODUCTION

Public benchmarks are central to evaluating time series foundation models (TSFMs), yet large pretraining collections are not always documented at the series level. Chronos, TimesFM, Moirai, and Time-MoE learn from diverse time series collections (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Shi et al., 2025). Recent studies identify overlap, unclear data partitions, and incomplete training data records as threats to reliable evaluation (Meyer et al., 2025; Li et al., 2026). If a benchmark continuation appeared in pretraining, low forecast error may reflect recall rather than generalization. We call the resulting overestimate of generalization *memorization inflation*.

This uncertainty also prevents data owners from reliably determining whether a released model learned from their series, making it harder to trace training data and audit their use. Dataset inference asks this question from model behavior after training, but its conclusions can depend on proxy labels and comparison data (Maini et al., 2024). Membership inference tries to tell whether a record appeared in training. Differences between datasets can make such tests look more successful than they are, and a positive result alone does not prove training data use (Maini et al., 2024; Zhang et al., 2025a). This gap motivates us to compare evidence collected after training with evidence designed in advance. We study two routes: analysis of a released model after training and randomized marking of the owner's data before release.

The second route sets up its comparison before training: for each host span, the owner randomly chooses which of two secret keys to embed. After training, the model is scored on both keyed versions of that same series, so the test does not rely only on comparisons between different datasets. Time series require a marking method designed for continuous inputs. TSFMs rescale each input and divide it into windows or patches before computing a loss (Ansari et al., 2024; Woo et al., 2024). A useful mark must retain a comparable strength across hosts, extend across training windows, and be reproducible from a secret key. These requirements motivate TimeCanary's sparse Fourier

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.

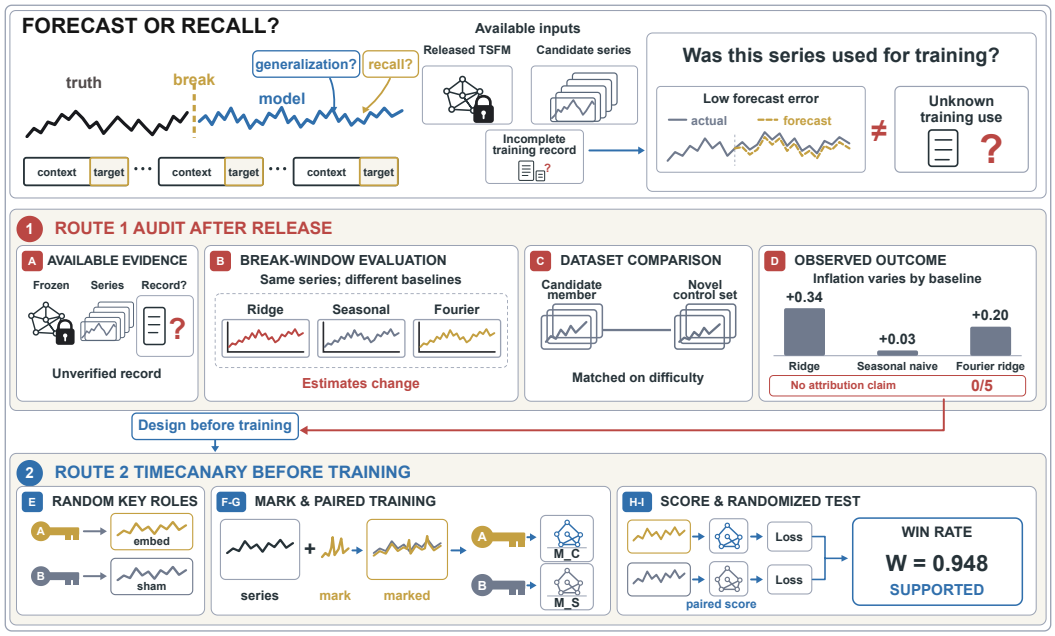

Figure 1: Forecast or recall. Low forecast error does not establish training use. A post-release audit gives baseline-dependent estimates but no attribution claim. TimeCanary instead randomizes key roles before training and tests a paired key preference; the shown win rate comes from the full-share Chronos experiment. Audit values are in Table 5.

construction, scaled to each host, and its comparison of model-specific losses on two keyed versions of the same host (Figure 1).

Building on this design, we make four contributions.

- We define memorization inflation at regime breaks, where the data pattern changes, and build an estimator based on the continuous ranked probability score (CRPS). In an audit of a released model, the estimated inflation changes with the baseline forecaster. The benchmarks also have too few valid within-dataset comparisons to reach a reliable conclusion.
- We introduce **TimeCanary**, which marks continuous time series with sparse Fourier signatures scaled to each host and compares model-specific losses on two keyed windows.
- We pair two randomly assigned keys with a matched model whose key roles are reversed. This design cancels differences between the keys that exist before training and supports an exact randomization test.
- Across five of the six model families tested, TimeCanary provides consistent attribution evidence on the tested host populations. In a separate Chronos small experiment, the paired score also met its support rule with 62 of 250 eligible host spans marked.

Section 2 reviews prior methods. Sections 3 and 4 study attribution after training. Sections 5 and 6 present TimeCanary, its results, and their limits. Section 7 concludes the paper.

2 Related Work

Time series foundation models. Across the post-training audit and TimeCanary experiments, we consider Chronos and Chronos-Bolt (Ansari et al., 2024; Amazon, 2024), TimesFM (Das et al., 2024), Time-MoE (Shi et al., 2025), Moirai and Moirai-MoE (Woo et al., 2024; Liu et al., 2025), Toto (Cohen et al., 2025), and Lag-Llama (Rasul et al., 2023). These models learn from large collections covering many domains. Moirai's LOTSA release lists its source datasets (Salesforce AI Research, 2024), but our method does not require such a list. The post-training audit uses four checkpoints; TimeCanary covers nine checkpoints from six model families.

Membership inference and extraction. Membership inference uses model outputs to ask whether a record appeared in training (Shokri et al., 2017; Wu & Cao, 2025). In tests on large language models, these methods did little better than guessing in most settings (Duan et al., 2024). Even stronger tests cannot by themselves show which dataset supplied a record (Zhang et al., 2025b; Hayes et al., 2025; Zhang et al., 2025a). Dataset inference aggregates evidence across records (Maini et al., 2024), while time series work tests record and user membership in forecasters built for specific tasks (Johansson et al., 2025). Data extraction instead tries to make a model reproduce examples from its training data (Carlini et al., 2021). All these methods examine a model after training. Using their findings for attribution requires a reliable record of the training data, which an external auditor may not be able to verify.

Contamination auditing. A related approach searches directly for overlap between benchmarks and training data (Cheng et al., 2025). TSFMAudit examines six TSFM backbones and 187 datasets (Li et al., 2026), while Meyer et al. (2025) analyze TSFM evaluation leakage. Other work builds fresh time series benchmarks (Qiao et al., 2026) or marks text benchmarks (Sander et al., 2025). These studies help find overlap in test data or build cleaner tests. They do not study a mark placed in an owner's original time series with the paired, role-swapped test used here.

Canaries and watermarks: evidence designed before release. Other data types have canaries and watermarks too, but each task needs its own test of whether the data remain useful. Text traps and randomized watermarks study recall, detection power, and false positive control (Meeus et al., 2024; Wei et al., 2024; Weinberg, 2026). For text benchmarks, rephrasing methods check whether models can still perform the original task (Sander et al., 2025). For image datasets, ownership marks are assessed through validation accuracy, image distortion, and how visible the changes are (Bouaziz et al., 2025). Similarly, TimeWak measures generation quality and predictive utility alongside detectability and robustness for watermarked diffusion outputs (Soi et al., 2025). Other designs use indirect poisoning or audit leakage from synthetic text (Bouaziz et al., 2026; Meeus et al., 2025). Their quality measures are not directly comparable because the protected objects and downstream tasks differ. TimeCanary asks whether changing an owner's original series preserves forecast utility while leaving a keyed trace of training.

To our knowledge, no prior method plants such a trace in an owner's series to attribute its use in TSFM training. A TSFM rescales and tokenizes each series before computing a loss (Ansari et al., 2024; Woo et al., 2024). A mark of absolute amplitude loses its intended scale under window normalization, and a mark at one timestep can disappear under patch tokenization. Continuous numeric inputs also lack the repeated discrete sequence used by text watermarks. TimeCanary's Fourier construction, scaled to each host, addresses these constraints (§5.1).

3 Measuring memorization inflation

We use CRPS, a forecast error score, to estimate the *memorization inflation* of §1: the difference between forecasting a new pattern and recalling a continuation seen in pretraining. We check it on one public dataset pair and several matching tolerances, then apply it to a released model (§4). A separate negative log likelihood (NLL) score tests membership that we create and therefore know. Appendix B gives the tables.

The estimator. For a frozen backbone M and break window E, the *excess* is the CRPS of a baseline forecaster, called the *anchor*, minus the CRPS of M. The anchor is ridge regression fit only before the break, and excess is positive when the model beats it. CRPS is normalized within each window set (*arm*) by that set's median standard deviation before the break.

For a candidate member set E_{mem} and a novel control set E_{novel} matched on difficulty (Appendix B), we define inflation as

$$\text{inflation}(M) = \underset{E \in E_{\text{mem}}}{\text{median}}\, \text{excess}(M, E) \;-\; \underset{E \in E_{\text{novel}}}{\text{median}}\, \text{excess}(M, E). \tag{1}$$

The median limits extreme values. A candidate signal needs a paired bootstrap 95% CI excluding 0.

Checks for misleading score differences. A *within dataset null* check runs the same statistic against every other series in the candidate member's dataset, where membership cannot differ. It sets the

floor a cross dataset inflation must clear and is used in §4.2. A separate *within reference* null holds one reference dataset fixed; only the baseline sensitivity analysis in §4.1 uses it. Appendix B gives both procedures.

Checks on public benchmarks and constructed membership. First, at the recorded matching tolerance, all four tested checkpoints return a null on the matched ETTh1 and ETTh2 pair, which has no known membership difference. This check covers one series, horizon, and pair; every null floor exceeds its point estimate (Table 2). Second, the smallest Time-MoE checkpoint raises no candidate at any tested matching tolerance. Separately, an NLL score detects membership that we create. Its stronger setting separates members from nonmembers perfectly, through better member scores and worse nonmember scores (Table 4). These limited checks do not fully validate the CRPS estimator, but the NLL result shows that known membership can be detected, as TimeCanary tests with randomized keys in §5.2.

4 Limits of attribution after training

4.1 The observational result changes with the baseline

The audit applies the excess score of §3 to Moirai-1.0-R-large. Australian electricity demand is the candidate member because Moirai's LOTSA registry lists it (Salesforce AI Research, 2024). ETTm1, ETTm2, ETTh2, and exchange rate are matched controls not listed as members; ETTh1 supplies the within reference null. These registry labels do not verify actual training use. We compare three baselines fit only before the break: ridge has no seasonal term, seasonal naive repeats the last season, and Fourier ridge adds sine and cosine terms for the period.

The planned tests use a within dataset floor for each baseline. The candidate member stays above this floor under every baseline. Yet its estimated inflation under seasonal naive is about one tenth of its ridge value (Appendix C, Table 5). This large change in size means that a result from one baseline alone is not enough to justify a claim about training data use.

4.2 Too few valid series pairs for a reliable audit

For each dataset, the audit subtracts its within dataset null floor under each baseline and keeps the smallest gap above that floor, so choosing a favorable baseline cannot create a positive result. Only the candidate member has a positive smallest gap. Benjamini and Hochberg false discovery rate control (Benjamini & Hochberg, 1995) across all five audited datasets returns zero discoveries (Table 7; decision rule, correction level, and configuration in Appendix C). The point estimate singles out the registry member, but too few valid null pairs remain for the corrected test to declare a discovery at any effect size.

The result changes with the baseline, and requiring agreement across baselines leaves too little power. Thus, on these benchmarks the audit cannot justify a memorization claim in either direction. Its reach beyond them is untested. Adding windows cannot fix the shortage of valid series pairs, so TimeCanary creates its own randomized null (§5).

5 TimeCanary: Marking and inference

The audit in §4 has too few valid null pairs in the tested benchmarks. TimeCanary randomly assigns key roles before release, but still needs enough clusters to test (Appendix E). A *host* is a selected span of an original series; one series can supply several hosts. Algorithm 1 summarizes the test, and Figure 2 shows the full design.

Algorithm 1 TimeCanary attribution procedure.

1: For each host h, draw unused keys k_A, k_B; a fair coin assigns embedded and sham roles.
2: For each seed r, fit M_C on embedded keys and M_S on sham keys; fit clean M_0.
3: For each (h, r), score both keyed windows with M_C, M_S to form both scores.
4: Average seeds by host; compute cluster gap G, host win rate W, and sign-flip p_{rand}.
5: Apply the support rule and the checks required for each score.

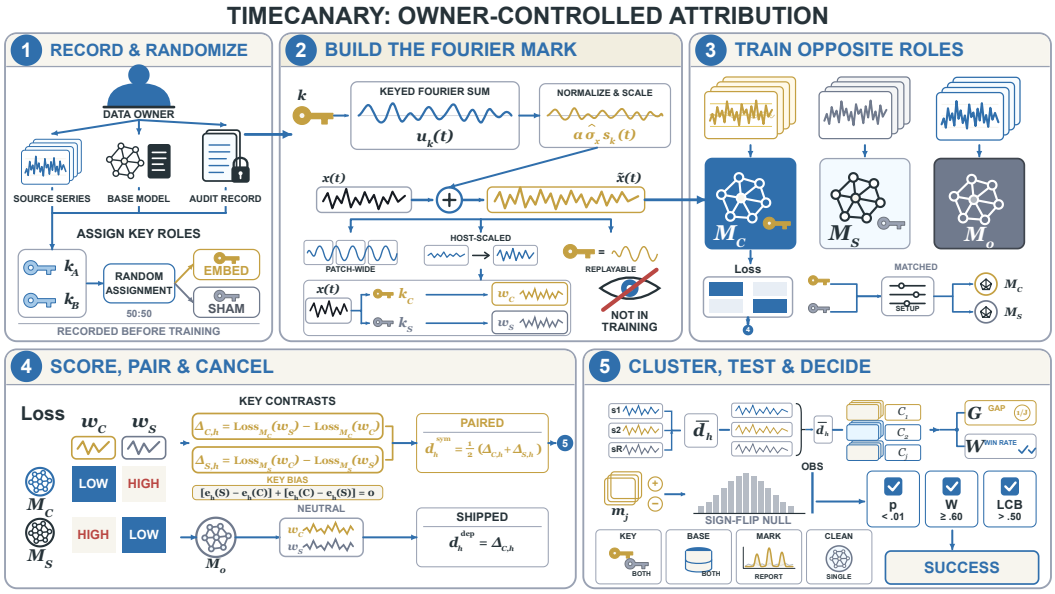

Figure 2: TimeCanary protocol. (1) The owner records the series, base model, audit record, and random key roles before training. (2) The embedded key makes a Fourier mark scaled to each host, and both keyed windows are kept for scoring. (3) The shipped model and its matched model train on opposite roles; the clean model is used only for a check. (4) Both models are scored on both windows. Averaging the two contrasts gives the paired score and cancels key difficulty; the single model score uses the shipped model alone. (5) Seed scores are averaged within hosts, hosts are grouped into physical clusters, and the sign flip test, win rate, interval, and required checks decide support.

5.1 Fourier mark construction

A secret nonnegative integer key k seeds frequencies f_j and phases ϕ_j for a sparse Fourier signature. For a host x of length L, define the raw Fourier sum u_k, its standardized signature s_k, and the marked series by

$$\begin{aligned} u_k(t) &= \sum_{j=1}^{n_{\text{comp}}} \sin(2\pi f_j t + \phi_j), \quad t = 0, \dots, L-1, \\ s_k(t) &= \frac{u_k(t) - \bar{u}_k}{\hat{\sigma}_{u_k}}, \\ \tilde{x}(t) &= x(t) + \alpha \hat{\sigma}_x s_k(t), \qquad \alpha = 0.12 \text{ by default}, \end{aligned} \tag{2}$$

Here $\bar{u}_k$ and $\hat{\sigma}_{u_k}$ are the sample mean and standard deviation of the raw sum on the marked interval, so s_k has zero mean and unit standard deviation; $n_{\text{comp}} = 3$ by default, and $\hat{\sigma}_x$ is the host standard deviation. Rejection sampling inside the Nyquist limit avoids near harmonics among the selected components. The same key reproduces the signature at any length (Appendix E).

Fourier components spread the mark across patches, scaling by the host preserves its relative amplitude under window normalization, and the chosen frequencies leave a key specific pattern for the matched filter survival diagnostic. Equation (2) adds the mark directly to the host's time series values, without first transforming the whole series into the frequency domain. The secret key determines the mark's frequencies, so an unmarked host does not reveal them. The marking method is public, however, so a trainer could drop marked spans (Appendix G.2); resampling can also change the frequency grid (Appendix G.1).

5.2 Two attribution scores

Comparing marked windows with windows the model did not see is hard to interpret because the marked windows are also training members. The mark is isolated by a *within model* comparison

between two candidate marks of the *same* host, only one of which the model trained on. At marking time the data owner generates two exchangeable candidate keys for each host and randomly chooses which one to embed (§5.4). The unused key is the *sham* key. This random choice lets the test ask whether the model favors the key it saw in training. For host h and one fixed training seed, let w_C be the scoring window carrying the embedded key and w_S the window carrying the sham key. For model M, let $\ell_M(w)$ be its recorded loss on scoring window w: NLL for the tested families other than Chronos-Bolt, which uses normalized pinball loss on forecast quantiles (Appendix F). Lower loss means a better fit. The single model score is

$$d_h^{\text{dep}} = \ell_{\text{shipped}}(w_S) - \ell_{\text{shipped}}(w_C), \tag{3}$$

positive when the shipped model favors its training key. No base model term appears, so an auditor with the shipped model and both keys can compute the score. In paired model experiments, it is the secondary score. Declaring support also requires diagnostic checks, including clean training neutrality, which needs a model trained on unmarked data (Table 1).

Differences between the keys that exist before training can make one key easier to fit and cause clean training neutrality to fail (as in the initial unfrozen single model experiment in §6). The *paired model score* adds a second model. Per seed, M_C is the shipped model trained on the embedded key, M_S the matched model trained on the sham key, and M_0 the clean model. It averages one trained key preference from each marked model:

$$\begin{aligned} \Delta_{C,h} &= \ell_{M_C}(w_S) - \ell_{M_C}(w_C), \\ \Delta_{S,h} &= \ell_{M_S}(w_C) - \ell_{M_S}(w_S), \\ d_h^{\text{sym}} &= \tfrac{1}{2}\big(\Delta_{C,h} + \Delta_{S,h}\big). \end{aligned} \tag{4}$$

Each delta is positive when its model favors its training key. Because M_C is shipped, $d_h^{\text{dep}} = \Delta_{C,h}$. Let $e_h(k)$ denote the part of the loss due to key k being easier or harder to fit, shared by the two models. It enters the two contrasts with opposite signs:

$$\Delta_{C,h}^{\text{key}} = e_h(S) - e_h(C), \qquad \Delta_{S,h}^{\text{key}} = e_h(C) - e_h(S), \qquad \Delta_{C,h}^{\text{key}} + \Delta_{S,h}^{\text{key}} = 0.$$

Thus the paired score removes shared key difficulty, however large it is. Other model preferences need not cancel for one realized key pair, but their average is zero under randomized key roles. Appendix E gives the full score argument. The paired score is the main test but needs M_S; it cannot be computed from the shipped model alone. The single model score uses the shipped model and both keys, but cannot remove a key preference present before training. It is therefore secondary and needs a clean model for its diagnostic check. Thus, access to the shipped model alone is not enough to apply either full support rule.

5.3 Randomization test and decision rule

A *cluster* contains hosts from one dataset and one block of consecutive time steps. We group channels from that block so the test does not count them as independent evidence. Blocks do not overlap. Hosts within one context length of a block edge are removed so adjacent blocks cannot share information (Appendix E). For each retained host h and seed r, let $d_{h,r}$ denote either attribution score. With R seeds and J clusters of hosts $\mathcal{H}_j$ ($N = \sum_j |\mathcal{H}_j|$ hosts), define

$$\bar{d}_h = \frac{1}{R}\sum_{r=1}^{R} d_{h,r}, \quad m_j = \frac{1}{|\mathcal{H}_j|}\sum_{h\in\mathcal{H}_j} \bar{d}_h, \quad G = \frac{1}{J}\sum_{j=1}^{J} m_j,$$

$$W = \frac{1}{N}\sum_{j=1}^{J}\sum_{h\in\mathcal{H}_j} \left[\mathbf{1}(\bar{d}_h > 0) + \tfrac{1}{2}\mathbf{1}(\bar{d}_h = 0)\right].$$

The attribution gap G weights physical clusters equally. The win rate W counts hosts after seed averaging, with ties counting half. Gaps are compared only within a model family, while win rates are compared across families.

Under the null, the model scores the two marked versions exchangeably. The one sided alternative is lower loss on the embedded key version. Simultaneously reversing all recorded host assignments

within cluster j sends m_j to $-m_j$. These cluster sign flips are exact under the sharp null that changing the recorded key assignment would not change the fitted model's scores. The earlier exploratory experiment in Appendix D, by contrast, did not randomly assign key roles, so it could not use this test. Let $\mathrm{LCB}_{0.95}(W)$ denote the lower endpoint of the 95% cluster bootstrap interval. The test statistic, randomization value, and planned support rule are

$$G(\boldsymbol{\epsilon}) = \frac{1}{J}\sum_{j=1}^{J}\epsilon_j m_j, \qquad \epsilon_j \overset{\text{iid}}{\sim} \{-1,+1\},$$

$$p_{\text{rand}} = \Pr_{\boldsymbol{\epsilon}}[G(\boldsymbol{\epsilon}) \geq G_{\text{obs}}]\,,$$

$$\mathcal{S}_{\text{stat}} = \begin{Bmatrix} p_{\text{rand}} < 0.01, & W \geq 0.60, \\ \mathrm{LCB}_{0.95}(W) > 0.50 & \end{Bmatrix}.$$

Support requires all three statistical conditions above and no failure in a check required for that score (Table 1). Both scores use the same clusters, omit hosts near block edges, and average repeated training runs in the same way. Each diagnostic uses a cluster level test with a $p < 0.10$ threshold. Base model neutrality uses a one sided test; the other diagnostics use two sided tests. For checks that can block support, this threshold favors catching a problem, although it can also reject a valid result.

5.4 Assignment and diagnostic checks

An earlier exploratory experiment exposed four design problems: unrandomized key assignment, pseudoreplication (treating correlated windows as independent), checkpoint dependence, and marking after window extraction (Appendix D). These problems led to the randomized paired test used here.

Randomization. For each host h, two keys $k_A(h)$ and $k_B(h)$ are drawn from disjoint integer ranges, neither previously used as a training key. A seeded fair coin z_h is generated and recorded *before* training. It assigns the embedded and sham roles according to

$$z_h \sim \mathrm{Bernoulli}(0.5), \qquad \big(k_{\text{emb}}(h), k_{\text{sham}}(h)\big) = \begin{cases} \big(k_A(h), k_B(h)\big), & z_h = 0, \\ \big(k_B(h), k_A(h)\big), & z_h = 1. \end{cases}$$

Thus each candidate key is embedded with probability $1/2$, while the other serves as the sham. This recorded assignment makes the sign flip test of §5.3 exact.

All four diagnostic checks are computed for each result (Table 1).

Table 1: Diagnostic checks and their effect on each score. A failed check makes a score unusable ("invalid"), denies attribution support ("null"), or leaves the decision unchanged ("report only").

diagnostic check	purpose	paired model score	single model score
key isolation	no preference between two keys unused in training	invalid if failed	invalid if failed
base model neutrality	base checkpoint has no key preference	null if failed	null if failed
mark visibility	mark creates a measurable signal	report only	report only
clean training neutrality	clean model M_0 has no key preference	canceled by design	invalid if failed

Interpreting the diagnostics. Mark visibility asks whether the mark carries a signal that a model can learn. It does not test whether someone can find the mark without knowing its key (Appendix G.2). Using visibility to block support would restrict support to marks the model cannot learn. It affected only the initial frozen experiment and was reported as a visibility measure thereafter (§6). Clean training neutrality can invalidate the single model score but not the paired model score because eq. (4) cancels the same key difficulty. Key isolation is therefore the only check that can invalidate the paired model score. It tests the trained models with two new keys; base model neutrality instead tests the original key pair on the untouched checkpoint. Appendix F gives the thresholds, operating point, power analysis, experiment sequence, and reported values (Table 10).

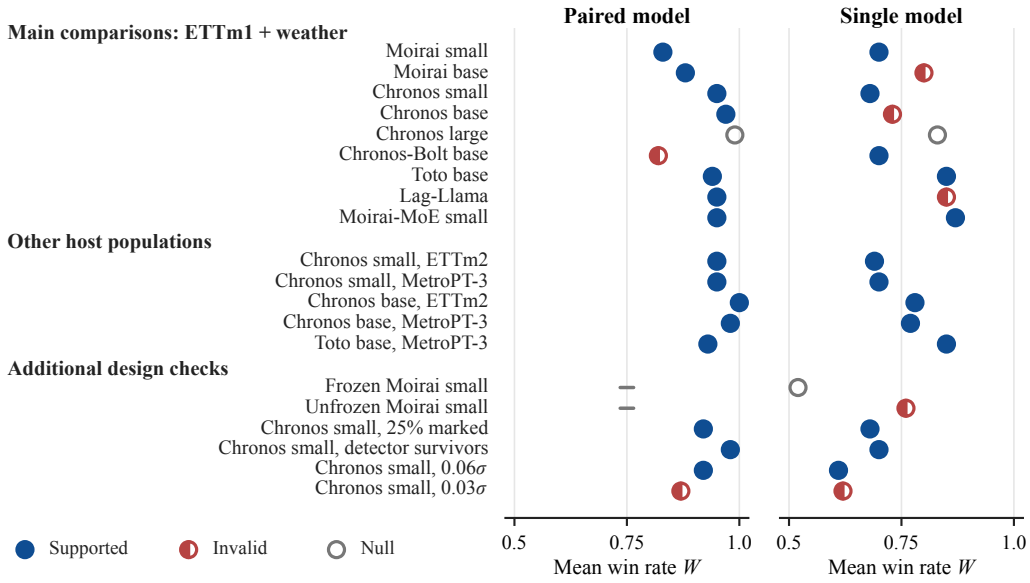

Figure 3: Mean win rates (paired left, single right). The dot style shows the final decision after the required checks. The two dashes in the paired panel mark early Moirai small experiments for which the matched model was not trained. Win-rate means and decisions are in Appendix F, Table 9.

6 Attribution Results

Figure 3 shows the paired and single model results from twenty experiments in separate panels; the full table is in Appendix F (Table 9). Fifteen used hosts pooled from ETTm1 and weather, three used MetroPT-3, and two used ETTm2 alone. Each host layout was fixed before training. These experiments test whether the result carries across model backbones. They do not separate ETTm1 from weather within the pooled experiments.

The paired model score supports attribution at the fixed operating point. The primary score of §5.2 was supported in fifteen of the eighteen paired model experiments, with every required diagnostic check passing in those fifteen; key isolation failed in two and base model neutrality in one (Table 9). The eighteen paired experiments together cover nine checkpoints from six TSFM families on three host populations.

To test whether every host had to be marked, a separate Chronos small experiment marked 62 of the 250 eligible host spans (about one quarter). The paired score still met its support rule ($W = 0.92$), and all required checks passed (Table 9). This result applies to this setting; Figure 4 shows the other tested marking shares, and Appendix F.4 gives the details.

The single model score is effective but depends on its diagnostic checks. The score in eq. (3), judged with all of its required checks, was supported in thirteen of the twenty experiments. The statistical criteria were met in the unfrozen single model Moirai, paired Moirai base, and Lag-Llama experiments, but the clean training neutrality check failed, so their single model results were invalid. The single model key isolation check failed for Chronos base and Chronos small at 0.03σ. Base model neutrality failed for Chronos large. The paired model score remained supported for paired Moirai base and Lag-Llama because it cancels the same key difference (§5.2). It also remained supported for Chronos base because the paired score's own key isolation check passed. Toto has more overlap between training and scoring windows than full share Chronos, so its larger values do not show a stronger mechanism. Full diagnostic results appear in Appendix F.

Updating the backbone weights was necessary under the tested settings. The initial Moirai experiment kept the backbone weights fixed, trained for 3000 steps, and failed the original mark visibility condition, but its statistical result was also null ($p = 0.286$). A single seed study selected the training and amplitude settings used later. Nine additional marked settings completed the grid after the main experiment sequence. They did not affect that choice (Table 11; Figure 5). The completed

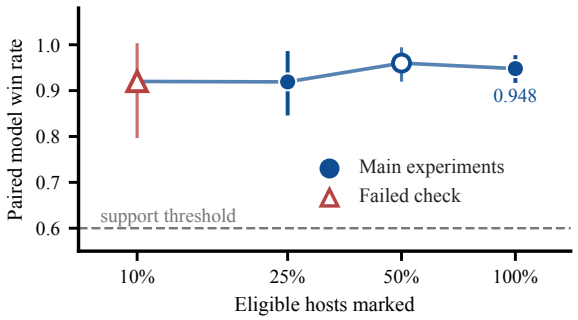

Figure 4: Paired model win rate by marked share. Filled blue circles show the main experiments; the open 50% circle is a separate check, and the red triangle failed key isolation. Vertical lines show 95% intervals. The line connects the observed means, not a fitted trend. Details are in Appendix F.4.

grid found little change when the encoder weights stayed fixed, whereas the effect increased with training steps when the weights could update. Later attribution experiments allowed the model weights to update and used 8000 steps. All but the two reduced-amplitude experiments used 0.12σ. This finding describes the tested settings rather than a rule that must hold for every model.

For an owner to use the mark, its effect on forecasting also matters. At 0.12σ, three of four dataset and forecaster checks had 95% intervals for increased MAE above zero; only ridge on ETTm1 had an interval that included zero. The largest estimated increase was 9.2% for ridge on weather ($[2.9, 16.7]\%$). At 0.25σ or above, all four checks showed worse forecast error.

A separate exploratory study with one training seed tested models trained on transformed marked spans. The single model score still met the full support rule after 8-bit quantization of those spans or noise at half the mark's scale, but not after noise at the mark's full scale, standardizing each training window, or downsampling by two (Appendix G).

The evidence concerns continued pretraining under the tested settings and host populations. Training drew an owner-pool batch with probability 0.8 and used that pool every tenth step regardless of the draw (Appendix F.1). The quarter marked experiment changed the share marked within the owner pool; we did not test a smaller owner share of the full training mix. Thus the quarter marked result tests how many of the owner's hosts carry a mark, not what happens when owner batches become rare in training. The pooled results do not separate ETTm1 from weather, and we did not test checkpoints beyond those reported, training from scratch, or adapter training. A key-blind boosted-tree detector flagged 64.8% of marked spans (95% interval: 58.0% to 70.5%) at a 2.4% false positive rate. After the flagged spans were replaced with clean ones, the single model score was supported on 88 marked hosts. Other detectors, dropping without clean replacements, and targeted signal removal were not tested with the full attribution rule (Appendix G.2). A supported single model decision also needs a clean model M_0 for its diagnostic check; the paired score needs a matched model M_S.

7 Discussion and Conclusion

On these benchmarks, our audit of unmarked data changed with the baseline forecaster and had too few valid null pairs to settle training use (§3, §4). TimeCanary instead places a randomized mark before release to create a controlled comparison. The recorded roles fix which key each model saw in training. The matched model reverses the key roles, canceling differences in key difficulty shared by the two models. This reduces the risk of treating an easier key as evidence that the model learned from the marked data. Required checks look for preferences that could arise without learning that key; they ruled out some otherwise positive results. A strong score alone is therefore not enough.

Future work should reduce forecast cost, test how marks survive data changes and removal, and make the full test possible with only a released model. It should also examine training methods not covered here. Within the tested settings, TimeCanary shows that a mark placed before release can provide evidence that a model learned from the series.

AI USE STATEMENT

In this work, we used generative AI tools only to assist with implementing methods in software, editing code, and polishing the manuscript for clarity and readability. We did not use these tools to generate, clean, or reformat datasets; develop theoretical models or mathematical claims; formulate hypotheses; design or evaluate research methodology or experiments; write proofs; or interpret results. Translation and qualitative or thematic analysis were not applicable. The authors reviewed every change produced with this assistance, tested the resulting code, and take responsibility for the final paper and accompanying artifacts.

ETHICS STATEMENT

The canary construction lets a data owner mark their own data and later provide the recorded coin result and keys for a controlled audit. The full support rules also need the matched or clean reference model and the base checkpoint. The construction is designed only for data the owner controls. No experiment here touches another party's data; every real dataset used (ETTh1, ETTh2, ETTm1, ETTm2, weather, exchange rate, Australian electricity demand, and MetroPT-3 (Davari et al., 2021)) is a public benchmark and every checkpoint audited or trained is a public release. No human subjects are involved. At the operating amplitude the mark is visible to the model it is embedded in, and standard forecast checks found increased forecast error in three of four tested settings, with the largest estimated increase at 9.2% (Appendix G.1). A trainer who knows the marking scheme but not the key can flag most marked spans and drop them (Appendix G.2). A practical limit is that resampling the marked data can defeat attribution in the tested setting (Appendix G.1), so a result with no detected mark does not show that the training data were unused.

REPRODUCIBILITY STATEMENT

The supplement includes anonymized TimeCanary code for mark construction, keyed scoring, four diagnostic checks, and cluster inference. It also includes runners for continued pretraining and attribution in fifteen Moirai and Chronos-T5 experiments. On publication, we will release the other four families' runners, the CPU test suite, launch scripts, dated protocols and amendment logs, manifests, console logs, results, follow-up checks after failed required checks, and scripts to regenerate the tables and figures. Once released, these materials will let readers trace each reported result to its settings and outputs.

REFERENCES

Amazon. Chronos-Bolt Base. Hugging Face model release, 2024. URL https://huggingface.co/amazon/chronos-bolt-base.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=gerNCVqqtR.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 1995. doi: 10.1111/j.2517-6161.1995.tb02031.x.

Wassim Bouaziz, Nicolas Usunier, and El-Mahdi El-Mhamdi. Data Taggants: Dataset ownership verification via harmless targeted data poisoning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wassim Bouaziz, Mathurin Videau, Nicolas Usunier, and El-Mahdi El-Mhamdi. Winter Soldier: Backdooring language models at pre-training with indirect data poisoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ORKhJBFepG.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650. USENIX Association, 2021.

Yuxing Cheng, Yi Chang, and Yuan Wu. A survey on data contamination for large language models, 2025. URL https://arxiv.org/abs/2502.14425.

Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean du Terrail, Anna-Monica Toon, Kan Wang, Stephan Xie, Zongzhe Xu, Viktoriya Zhukova, David Asker, Ameet S. Talwalkar, and Othmane Abou-Amal. This time is different: An observability perspective on time series foundation models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1698.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167. PMLR, 2024.

Narjes Davari, Bruno Veloso, Rita P. Ribeiro, and João Gama. MetroPT-3 dataset. UCI Machine Learning Repository, 2021. URL https://doi.org/10.24432/C5VW3R.

Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. Do membership inference attacks work on large language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=av0D19pSkU.

Jamie Hayes, Ilia Shumailov, Christopher A. Choquette-Choo, Matthew Jagielski, Georgios Kaissis, Milad Nasr, Meenatchi Sundaram Muthu Selva Annamalai, Niloofar Mireshghallah, Igor Shilov, Matthieu Meeus, Yves-Alexandre de Montjoye, Katherine Lee, Franziska Boenisch, Adam Dziedzic, and A. Feder Cooper. Exploring the limits of strong membership inference attacks on large language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2934.

Nicolas Johansson, Tobias Olsson, Daniel Nilsson, Johan Östman, and Fazeleh Hoseini. Privacy risks in time series forecasting: User- and record-level membership inference, 2025. URL https://arxiv.org/abs/2509.04169.

Hongkai Li, Shifeng Xie, Lefei Shen, Zhuo Li, Mouxiang Chen, Xiaobin Zhang, Han Fu, Jianling Sun, Xiaoxue Ren, and Chenghao Liu. TSFMAudit: Data contamination auditing in forecasting time series foundation models, 2026. URL https://arxiv.org/abs/2605.26161.

Xu Liu, Juncheng Liu, Gerald Woo, Taha Aksu, Yuxuan Liang, Roger Zimmermann, Chenghao Liu, Junnan Li, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Moirai-MoE: Empowering time series foundation models with sparse mixture of experts. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 38940–38962. PMLR, 2025.

Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. LLM dataset inference: Did you train on my dataset? In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3941. URL https://openreview.net/forum?id=Fr9d1UMc37.

Matthieu Meeus, Igor Shilov, Manuel Faysse, and Yves-Alexandre de Montjoye. Copyright traps for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35296–35309. PMLR, 2024.

Matthieu Meeus, Lukas Wutschitz, Santiago Zanella-Béguelin, Shruti Tople, and Reza Shokri. The canary's echo: Auditing privacy risks of LLM-generated synthetic text. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 43557–43580. PMLR, 2025.

Marcel Meyer, Sascha Kaltenpoth, Kevin Zalipski, and Oliver Müller. Rethinking evaluation in the era of time series foundation models: (Un)known information leakage challenges, 2025. URL https://arxiv.org/abs/2510.13654.

Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's TIME: Towards the next generation of time series forecasting benchmarks, 2026. URL https://arxiv.org/abs/2602.12147. Accepted to ICML 2026.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. Lag-llama: Towards foundation models for probabilistic time series forecasting. *arXiv preprint arXiv:2310.08278*, 2023. URL https://arxiv.org/abs/2310.08278.

Salesforce AI Research. LOTSA Data. Hugging Face, 2024. URL https://huggingface.co/datasets/Salesforce/lotsa_data/blob/main/README.md.

Tom Sander, Pierre Fernandez, Saeed Mahloujifar, Alain Durmus, and Chuan Guo. Detecting benchmark contamination through watermarking, 2025. URL https://arxiv.org/abs/2502.17259.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-MoE: Billion-scale time series foundation models with mixture of experts. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=e1wDDFmlVu. Spotlight.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017. doi: 10.1109/SP.2017.41.

Zhi Wen Soi, Chaoyi Zhu, Fouad Abiad, Aditya Shankar, Jeroen M. Galjaard, Huijuan Wang, and Lydia Y. Chen. TimeWak: Temporal chained-hashing watermark for time series data. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4604. Spotlight.

Johnny Wei, Ryan Wang, and Robin Jia. Proving membership in LLM pretraining data via data watermarks. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13306–13320, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.788.

Abraham Itzhak Weinberg. Subtle injection for ground-truth inference of LLM training data, 2026. URL https://arxiv.org/abs/2606.06502.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53140–53164. PMLR, 2024.

Hengyu Wu and Yang Cao. Membership inference attacks on large-scale models: A survey, 2025. URL https://arxiv.org/abs/2503.19338.

Jie Zhang, Debeshee Das, Gautam Kamath, and Florian Tramèr. Position: Membership inference attacks cannot prove that a model was trained on your data. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 333–345. IEEE, 2025a. doi: 10.1109/SaTML64287.2025.00025.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Yang, and Hai Li. Min-K%++: Improved baseline for pre-training data detection from large language models. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=ZGkfoufDaU. Spotlight.

A READER GUIDE

The appendix follows the paper's argument rather than the order in which the experiments were run. Appendix B checks the estimator used after training, and Appendix C gives the observational audit and its power limit. Appendix D records the early exploratory experiment only as motivation for the final design. Appendix E gives the TimeCanary method, Appendix F reports the experiments, and Appendix G examines their practical limits. The accompanying artifact contains complete randomization records, seed manifests, results for each audit setting, and implementation checks. Those records are not repeated here because they do not change the scientific interpretation.

Across the TimeCanary appendices, M_C denotes the model trained on the embedded key, M_S its matched counterpart trained on the other candidate key, and M_0 the model trained on unmarked data. For host h, w_C and w_S are the two keyed scoring windows. The single model score d_h^{dep} is defined in Eq. (3), and the paired model score d_h^{sym} is defined in Eq. (4). All confidence intervals are 95% intervals unless stated otherwise.

B ESTIMATOR VALIDATION

This appendix asks whether the estimator used after training behaves sensibly in three settings with different purposes. The public benchmark comparison should remain null, the matching tolerance should not determine the conclusion, and constructed membership should become detectable when exposure is strong. The checks support those limited expectations, but they do not by themselves validate attribution after training on naturally occurring membership labels.

B.1 PUBLIC BENCHMARK NULL CHECK

The estimator pairs each candidate window with its closest match from a distinct held out dataset. The matching space contains normalized volatility before the break, break magnitude, and signal to noise ratio. Aggregation uses the median because excess loss over an anchor is heavy tailed. The within dataset control applies the same calculation to series whose membership status cannot differ and uses the largest absolute inflation as the null floor.

Table 2: Public benchmark null check on ETTh1 and ETTh2, using the OT series at horizon 96 and the recorded matching tolerance. Every interval includes zero, and every point estimate remains below its backbone specific null floor. The Floor pairs column counts matched windows in the within dataset comparison that sets the floor, not the number of null comparisons.

Backbone	Inflation	95% CI	Pairs	Null floor	Floor pairs
Chronos-Bolt Base	$+0.0101$	$[-0.0608, +0.1311]$	145	0.3544	14
TimesFM 2.0 (500M)	$+0.0878$	$[-0.0534, +0.1569]$	146	0.1970	14
Moirai-1.0-R-large	-0.0485	$[-0.1400, +0.0606]$	142	0.3376	14
Time-MoE (50M)	-0.0132	$[-0.1057, +0.0774]$	144	0.3024	116

Table 2 gives the expected null result for two public benchmarks with no known membership difference. The check uses proxy labels on one dataset pair, one horizon, and one series. It therefore does not establish whether any of these models memorized either benchmark.

B.2 SENSITIVITY TO MATCHING TOLERANCE

Table 3: Tolerance sensitivity for Time-MoE on the same ETTh1 and ETTh2 comparison. The conclusion is null at every tolerance, and each estimate remains below its corresponding null floor. The Floor pairs column counts matched windows in the within dataset comparison that sets the floor; at tolerance 1.0, that comparison contains only two windows.

Tolerance	Inflation	95% CI	Pairs	Null floor	Floor pairs
1.0 (loose)	$+0.0191$	$[-0.0646, +0.1088]$	164	0.4184	2
0.5 (recorded)	-0.0132	$[-0.1057, +0.0774]$	144	0.3024	116
0.3 (tight)	$+0.0003$	$[-0.0732, +0.0845]$	104	0.3123	91

The median estimate remains near zero across the checked tolerances, but the floor at tolerance 1.0 rests on only two matched windows. An earlier version based on the mean was not stable. It reported an inflation of $+0.393$ with interval $[+0.063, +0.822]$ on the same backbone. The within dataset control rejected that apparent signal because its null floor was 0.69. Replacing the mean with the median removed the false positive at the score level, while the control remains necessary for unanticipated failures. The rule is therefore simple: a cross dataset estimate is read only when it exceeds the same backbone's within dataset null floor.

B.3 CONSTRUCTED MEMBERSHIP CHECK

The first two checks contain no known membership difference. We therefore fine tuned Moirai-1.1-R-small on 50 synthetic member series and held out 50 independently generated series from the same distribution. Each series was scored under the base and fine tuned checkpoints. The per series score was the reduction in NLL after fine tuning, and the constructed-membership gap compared its mean between members and nonmembers.

Table 4: Constructed membership detection under two fine tuning configurations. Each configuration changes several training quantities, so the comparison does not isolate one factor.

Fine tuning setting	Gap	95% CI	AUROC
Frozen encoder, 3000 steps, learning rate 10^{-4}, member share 0.5	$+0.302$	$[+0.110, +0.490]$	0.628
Unfrozen encoder, 8000 steps, learning rate 3×10^{-4}, member share 0.8	$+6.86$	$[+6.13, +7.66]$	1.000

The gentler configuration produces a modest but detectable gap. Members still deteriorate relative to the base checkpoint, but they deteriorate less than nonmembers. The stronger configuration produces perfect separation because members improve while nonmember performance becomes much worse. The contrast therefore demonstrates detectability at two joint configurations, but it does not isolate memorization from forgetting or establish a realistic operating point.

C OBSERVATIONAL AUDIT AFTER TRAINING

This appendix separates two questions. The first is whether the estimated inflation changes with the baseline forecaster, called the anchor. The second is whether a result that must hold across all anchors can produce a small enough probability value. The answer to the first question is yes. The answer to the second is no for the five audited datasets because their within dataset null pools are too small.

C.1 SENSITIVITY TO THE BASELINE FORECASTER

The audit compares the model with three anchors: ridge regression, seasonal naive forecasting, and Fourier ridge regression. Fourier ridge receives the same period information as the seasonal method

while retaining ridge extrapolation. It therefore tests whether the large change between the first two anchors can be explained by access to periodic structure.

Table 5: Australian electricity demand under the three baselines in the fixed-period audit. Inflation and its within dataset null floor use the same Moirai-1.0-R-large checkpoint and 2194 matched window pairs. The floor is the largest absolute inflation among valid comparisons between series in this dataset.

Baseline	Inflation	Within dataset floor	p
Ridge	$+0.3366$	0.2018	0.333
Seasonal naive	$+0.0316$	0.0169	0.333
Fourier ridge	$+0.2038$	0.0670	0.333

The seasonal naive estimate is 0.094 times the ridge estimate. The separate earlier comparison used another checkpoint and context length. Its saved per-cell outputs are not available here, so it is not used as evidence for this audit.

Table 6: Fourier ridge anchor with Moirai-1.0-R-large and ETTh1 as the reference. No dataset clears the reference null floor of 0.2418. Normalized MSE is an independent predictability measure.

Dataset	Registry role	Normalized MSE	Inflation	Matched pairs
Australian electricity demand	member	0.077	$+0.2038$	2194
ETTm1	nonmember	0.734	-0.0141	604
ETTh2	nonmember	1.088	-0.0779	166
ETTm2	nonmember	1.210	$+0.0744$	689
Exchange rate	nonmember	3.007	$+0.0762$	68

The candidate member is much easier to forecast than every nonmember in Table 6. Across the five datasets, predictability and estimated inflation have Spearman correlation $\rho = -0.100$. Its exact two sided permutation value is $p = 0.950$. Under this anchor the small set shows no steady relation, so the run gives no support for a predictability explanation, and the small sample prevents ruling one out. The member's predictability still lies outside the nonmember range. A fitted line at that point would extend beyond the observed nonmember data, so we do not interpret it.

C.2 Decision rule across all anchors

For each dataset and anchor, let Δ denote excess loss over the anchor and let the floor be the largest corresponding within dataset null value. The audit requires the smallest value of $\Delta - \text{floor}$ across the three anchors to be positive. It also requires a Benjamini and Hochberg discovery at $q = 0.1$ under every anchor. Using the smallest gap prevents a positive result from being created by selecting the most convenient anchor.

The reported run uses fixed seasonal periods of 48 for Australian electricity demand, 96 for ETTm1 and ETTm2, 24 for ETTh1 and ETTh2, and 1 for Exchange rate. Earlier automatically inferred periods produce the same zero discovery conclusion. All fifteen dataset and anchor results for both configurations are provided in the accompanying artifact.

Table 7: Audit summary across all anchors. The candidate member is the only dataset with a positive smallest gap, but no dataset is an FDR discovery because the null pools are too small.

Dataset	Smallest gap	Limiting anchor	Pairs	Null pairs	p
Australian electricity demand	$+0.0148$	seasonal naive	2194	2	0.333
ETTm1	-0.1678	ridge	604	3	0.750
ETTm2	-0.2305	ridge	689	3	1.000
ETTh2	-0.1997	Fourier ridge	166	3	0.750
Exchange rate	-0.2634	ridge	68	3	0.500

The registry listed member is the only dataset whose point estimate clears every anchor floor. This does not become a discovery. With only two valid null comparisons for that dataset, the smallest possible one sided empirical value is

$$p_{\min} = \frac{1}{2+1} = \frac{1}{3} \approx 0.333.$$

At $q = 0.1$ across five datasets, even the most lenient Benjamini and Hochberg critical value is 0.10. The available null comparisons therefore cannot produce a small enough probability value, regardless of the effect size or the 2194 matched windows. More windows improve the estimate of Δ, but only more valid series pairs can provide smaller null probability values. This is why the audit has too little power at any effect size under the planned procedure.

The forecasts behind every reported number in this appendix and §4 come from the Moirai-1.0-R-large checkpoint (Woo et al., 2024), the release whose LOTSA pretraining corpus is documented. An earlier cache, keyed under a Moirai-2-R-large name that no published checkpoint carries, had mixed two models when the requested model failed to load: Moirai-1.1-R-large for the controls and Moirai-2.0-R-small for the member. We discarded that cache and reran every audit on one checkpoint. The run script in the accompanying artifact checks the loader identity of every cell and the cache manifest before any audit runs.

The candidate member label for Australian electricity demand is recorded in the source registry. The nonmember labels for the ETT datasets follow explicit exclusion from LOTSA. The Exchange rate label is weaker because it rests on absence from the public registry rather than an explicit exclusion statement. This registry uncertainty is separate from, and additional to, the statistical power limit.

D Early exploratory experiment

This experiment preceded the final TimeCanary design. It is included because its failures motivated the randomized, cluster level protocol, whose score does not compare a base checkpoint with its trained descendant. None of its hypothesis tests contributes evidence in the main results.

D.1 Setup and planned result

The experiment used ETTh1 and Moirai-1.1-R-small. Continued pretraining ran for 800 gradient steps at learning rate 5×10^{-5} with a frozen encoder. The analysis used 120 nonoverlapping windows, divided equally between members and controls. At the planned marked share of 0.5, the member windows were divided equally between marked and unmarked examples.

The planned comparison asked whether marked members benefited more than unmarked members. It did not support that prediction. The NLL gap was $+0.0384$, the uncalibrated permutation value was $p = 0.3362$, and the AUROC was 0.5178 with interval $[0.3733, 0.6656]$. The canary did not add a detectable membership signal at this operating point.

D.2 Why the early experiment could not establish attribution

Four design defects prevent an attribution interpretation.

1. **Key assignment was not randomized.** The trained and unused keys came from fixed, separated ranges. The paired test therefore assumed exchangeability that the design did not create.

2. **The analysis treated correlated windows as independent.** This problem is called pseudoreplication. The counted observations were adjacent time windows and channels from one series, not independent physical units. A synthetic check with six true clusters showed that treating 30 correlated windows as independent drove the permutation value at its numerical floor, while a cluster level test on the same data returned values near 0.02 to 0.03.

3. **The statistic required two checkpoints.** It compared the base model with its fine tuned descendant. A data owner holding only the shipped model could not compute it.

4. **The mark was applied after window extraction.** The early design assumed the trainer's context, stride, and alignment instead of embedding the mark in the continuous source series.

ETTh1 also supplies only 17 nonoverlapping blocks at the final block length, below the required minimum of 20 clusters. The experiment could not have met the later inference standard even if the four design defects had been absent.

D.3 Exploratory patterns that motivated redesign

Table 8: Selected results from the early exploratory experiment. The probability values are not validly calibrated because the analysis treated correlated windows as independent. The first row is the planned comparison.

Marked share	Comparison and seed	Gap	p	AUROC
0.50	Marked versus unmarked members, seed 7	$+0.0384$	0.3362	0.5178
0.25	Training key versus unused key, seed 7	$+0.1447$	$< 10^{-4}$	0.6800
0.50	Training key versus unused key, seed 7	$+0.1339$	$< 10^{-4}$	0.6056
0.50	Training key versus unused key, seed 17	$+0.1070$	$< 10^{-4}$	0.6067
0.50	Training key versus unused key, seed 23	$+0.1535$	$< 10^{-4}$	0.6444
1.00	Training key versus unused key, seed 7	$+0.1638$	$< 10^{-4}$	0.6475

The unused key comparison was positive at every checked marked share and at all three repeated training runs shown in Table 8. The reported values below 10^{-4} do not provide strong evidence: they match the numerical floor expected when correlated windows are treated as independent. The moderate AUROCs describe the same exploratory pattern but do not fix this validity problem.

A separate step count sweep held the learning rate and window count fixed. Its gaps were $+0.0634$, $+0.1339$, $+1.0201$, and $+3.1551$ at 400, 800, 3000, and 8000 steps, while AUROC increased from 0.5567 to 0.9978. This steady increase suggested that training was learning something about the marked windows. It did not show that the learned feature was specific to the assigned key.

The later randomized, cluster level design did not show the same frozen encoder pattern. At 3000 and 8000 steps, its gaps were $+0.0137$ and $+0.0067$. The mark placement, key assignment, and analysis also differed, so these results are not a direct repeat of the early sweep. That sweep motivated the redesign but does not provide attribution evidence.

E Canary construction and inference

This appendix explains why TimeCanary uses a Fourier signature scaled to each host, how the mark is embedded before training windows are chosen, and why the paired model score removes differences that exist between candidate keys. The final controls and experiment sequence are reported in Appendix F.

E.1 Fourier signature design

Equation (2) defines a raw sum of three sinusoids and standardizes that sum before host scaling. Their frequencies are drawn from the fixed band between 0.03 and 0.45 cycles per step. A candidate frequency is accepted only when its ratio to every previously accepted frequency stays more than 0.02 away from simple ratios with denominator at most four. This rule avoids near harmonics among the three components; it does not exclude common seasonal frequencies.

The Fourier construction provides three properties needed for time series foundation models. First, the signature spans the full marked interval, so dividing the series into patches cannot remove it by discarding one isolated timestamp. Second, scaling the unit variance signature by the host standard deviation gives the mark a comparable relative amplitude across series with different units and ranges. Third, a secret key reproduces exactly the same small set of frequencies and phases, which lets the owner construct both keyed scoring windows from the series retained for the audit.

These properties motivate the construction but do not guarantee that the mark survives every data transformation. Scaling by the host keeps normalization from changing the amplitude unpredictably. It does not ensure that a trained model retains the mark after per window standardization or resampling. The measured boundaries appear in Appendix G.

A matched filter provides a signal level diagnostic. It sums periodogram power at the known frequencies and divides by local background power. A value near one is expected when the keyed peaks are absent, and a larger value indicates that spectral energy remains at the keyed frequencies. This diagnostic is not the attribution statistic, has no role in the membership decision, and is used only for the robustness study.

E.2 Continuous series marking and physical clusters

A host is one selected, nonoverlapping span from a source channel, not the whole channel. The final design uses 512-step spans and 128-step training examples. Each source channel supplies as many complete spans as fit; a seeded shuffle selects at most 250 hosts before keys are assigned. Thus 250 hosts can come from far fewer than 250 distinct channels. Each experiment fixes its selected spans before training. The number of occupied clusters J can change when the selected spans fall in different dataset and time blocks, even when the host count remains 250.

The mark is embedded in a continuous source segment whose length is four times the training example length. Window extraction occurs afterward. The trainer therefore chooses crop positions, alignment, and stride without constraints from the marking procedure. This corrects the earlier design, which marked an already extracted window.

A cluster contains hosts from one dataset and one block of consecutive time steps. Channels recorded in that block are grouped, rather than counted as independent series. For example, ETT channels are measurements from the same physical instrument. Blocks do not overlap, and adjacent blocks are separated by at least one context length. Every study must contain at least 20 occupied physical clusters before inference proceeds.

The two candidate keys and each host's embedded key assignment are set before training. Repeated training runs share the same host layout and assignments. Their host level contrasts are averaged before one sign flip test and one cluster bootstrap are computed. Consequently, the number of training runs does not increase the number of independent units in the test.

The dated protocols and saved manifests document the assignments, but do not provide independent public proof of their timing. In a future external audit, an owner could post a hash of the protocol, host assignments, keys, coin results, and a long secret random value to an independent time-stamped record before release, then reveal these values for an auditor to check after training. We did not do this here.

E.3 Parts of the role swapped score

The two candidate keys can differ in difficulty even when neither was observed during training. The single model score cannot remove this difference because it scores only one trained model. The paired model score introduces the matched model with reversed key roles.

For host h, write the score of model M on the window carrying key k as

$$\ell_M(w_k) = b_{M,h} + e_h(k) + g_{M,h}(k) - \tau_{M,h}(k), \qquad k \in \{C, S\}. \tag{5}$$

The term $b_{M,h}$ is the model and host baseline. The term $e_h(k)$ is the difficulty of fitting key k, shared across models. The term $g_{M,h}(k)$ is a model specific preference that is not caused by learning this key. The term $\tau_{M,h}(k)$ is the loss reduction associated with learning the key during training. This equation is a way to divide the score into parts for the argument; it is not an additional fitted model.

Substituting Eq. (5) into the first role gives

$$\begin{aligned}\Delta_{C,h} &= \big[b_{M_C,h} + e_h(S) + g_{M_C,h}(S) - \tau_{M_C,h}(S)\big] \\ &\quad - \big[b_{M_C,h} + e_h(C) + g_{M_C,h}(C) - \tau_{M_C,h}(C)\big] \\ &= e_h(S) - e_h(C) + g_{M_C,h}(S) - g_{M_C,h}(C) + \tau_{M_C,h}(C) - \tau_{M_C,h}(S).\end{aligned} \tag{6}$$

Because M_C is the shipped model, $\Delta_{C,h} = d_h^{\mathrm{dep}}$. The same substitution for the reversed role gives

$$\begin{aligned}\Delta_{S,h} &= \big[b_{M_S,h} + e_h(C) + g_{M_S,h}(C) - \tau_{M_S,h}(C)\big] \\ &\quad - \big[b_{M_S,h} + e_h(S) + g_{M_S,h}(S) - \tau_{M_S,h}(S)\big] \\ &= e_h(C) - e_h(S) + g_{M_S,h}(C) - g_{M_S,h}(S) + \tau_{M_S,h}(S) - \tau_{M_S,h}(C).\end{aligned} \tag{7}$$

The baseline cancels inside each contrast. Averaging the two roles then yields

$$\begin{aligned}2d_h^{\mathrm{sym}} &= \underbrace{e_h(S) - e_h(C) + e_h(C) - e_h(S)}_{0} \\ &\quad + \big[g_{M_C,h}(S) - g_{M_C,h}(C) + g_{M_S,h}(C) - g_{M_S,h}(S)\big] \\ &\quad + \big[\tau_{M_C,h}(C) - \tau_{M_C,h}(S) + \tau_{M_S,h}(S) - \tau_{M_S,h}(C)\big].\end{aligned} \tag{8}$$

Key difficulty therefore cancels exactly, regardless of its size. Other model preferences need not cancel for one realized pair, but their average contribution is zero under the randomized key and model role symmetry. The remaining terms compare learning of the assigned training key with transfer to the other key. This specific key learning contrast is the attribution target.

E.4 Exact randomization inference

The sharp null states that the recorded key assignment has no effect on the fitted models' scores. Under that null, reversing one host's assignment swaps w_C and w_S while leaving the scoring functions fixed. The single model contrast changes sign, and both role contrasts in the paired model score change sign, so $d_h^{\mathrm{dep}} \mapsto -d_h^{\mathrm{dep}}$ and $d_h^{\mathrm{sym}} \mapsto -d_h^{\mathrm{sym}}$. The randomization action indexed by ϵ_j reverses every host assignment in $\mathcal{H}_j$ at the same time and therefore sends m_j to $-m_j$. Every fair assignment and its reversed version were equally likely. The test is therefore exact, conditional on the relative assignments inside each cluster.

This change used to form the null distribution differs from swapping the full construction. Swapping both window labels and model roles leaves d_h^{sym} unchanged and shows that the score does not depend on which candidate key happened to receive the embedded role. The randomization test swaps labels only. Keeping these two operations separate prevents the construction symmetry from being mistaken for the null distribution.

The two marked models use the same random choices. They share weight initialization, batch order, crop schedule, and background sampling, and differ only in which candidate key appears in the marked spans. The clean model M_0 uses an independent random stream and does not enter the paired score.

F Experiment sequence

F.1 Final settings and decision rule

The main pooled ETTm1 and weather design selects 250 eligible host spans. The ETTm2 and MetroPT-3 designs also select 250 hosts, though the selected spans occupy different numbers of

physical clusters (Table 10). Each study uses one checkpoint, one set of host assignments, and five training seeds. In the final design, each training step drew a whole batch from the owner host pool with a requested probability of 0.8, or from background data otherwise. Every tenth step used the owner pool regardless of the draw, and the achieved rate was recorded. The per-window score is NLL for the tested families other than Chronos-Bolt. Chronos-Bolt instead uses pinball loss on nine forecast quantiles over the 32-step target, averaged over target steps, summed over quantiles, and divided by the context's instance-normalization scale. Lower scores mean a better fit; attribution gaps are compared only within a model family. Every attribution experiment after the initial frozen study allows the model weights to update and uses 8000 gradient steps. The learning rate is 5×10^{-5} except for Lag-Llama. A separate learning-rate check selected 5×10^{-4} for its attribution run. The mark amplitude is 0.12 times the host standard deviation except in the 0.06σ and 0.03σ experiments.

Support requires a one sided randomization value below 0.01, a win rate of at least 0.60, and a 95% cluster bootstrap interval whose lower endpoint exceeds 0.50. A significant randomization value with an interval that still includes 0.50 is treated as a null result. Over the observed 94 to 104 clusters, the design has approximately 0.83 power for a true win rate near 0.70. It has less power for smaller effects.

Diagnostic checks use a failure threshold of 0.10. For key isolation, each host gets a second fair-coin assignment between two keys from ranges disjoint from the training keys. Neither key is used to train either marked model. The trained models score windows made from the same host span and scoring offset as the original pair, but with these unused keys. The paired and single model contrasts are each tested two sided; a detected preference invalidates that score. This checks for a key preference that training on the owner's marked data cannot explain. Base model neutrality instead scores the original key pair on the untouched checkpoint and yields a null when it fails. Mark visibility compares marked and unmarked windows and is reported without affecting the final decision. Clean training neutrality applies the single model score to a model trained on unmarked data. It can invalidate the single model score, but not the paired model score, because the role swap removes the same key difficulty. In the table below, *null* means the statistical rule was not met or base model neutrality failed; *invalid* means key isolation or clean training neutrality failed. Moirai small and Moirai base denote Moirai-1.1-R-small and Moirai-1.1-R-base.

Table 9: Win rates, 95% cluster bootstrap intervals, and attribution decisions by model and setup.

	paired model			single model		
model and setup	W	95% interval	decision	W	95% interval	decision
Main model comparisons (pooled ETTm1 and weather)						
Moirai small	0.83	[0.779, 0.876]	supported	0.70	[0.642, 0.757]	supported
Moirai base	0.88	[0.836, 0.915]	supported	0.80	[0.745, 0.842]	invalid (clean)
Chronos small	0.95	[0.921, 0.973]	supported	0.68	[0.628, 0.732]	supported
Chronos base	0.97	[0.945, 0.988]	supported	0.73	[0.677, 0.777]	invalid (key)
Chronos large	0.99	[0.980, 1.000]	null (base)	0.83	[0.787, 0.866]	null (base)
Chronos-Bolt base	0.82	[0.772, 0.863]	invalid (key)	0.70	[0.642, 0.746]	supported
Toto base	0.94	[0.909, 0.966]	supported	0.85	[0.809, 0.886]	supported
Lag-Llama	0.95	[0.920, 0.976]	supported	0.85	[0.800, 0.893]	invalid (clean)
Moirai-MoE small	0.95	[0.925, 0.977]	supported	0.87	[0.827, 0.908]	supported
Other host populations						
Chronos small, ETTm2	0.95	[0.924, 0.979]	supported	0.69	[0.626, 0.747]	supported
Chronos small, MetroPT-3	0.95	[0.923, 0.976]	supported	0.70	[0.639, 0.752]	supported
Chronos base, ETTm2	1.00	[0.987, 1.000]	supported	0.78	[0.727, 0.827]	supported
Chronos base, MetroPT-3	0.98	[0.967, 0.996]	supported	0.77	[0.719, 0.821]	supported
Toto base, MetroPT-3	0.93	[0.895, 0.957]	supported	0.85	[0.807, 0.895]	supported
Additional design checks (pooled ETTm1 and weather)						
Frozen Moirai small	n/a	n/a	not defined	0.52	[0.457, 0.577]	null
Unfrozen Moirai small	n/a	n/a	not defined	0.76	[0.702, 0.809]	invalid (clean)
Chronos small, 25% marked	0.92	[0.850, 0.982]	supported	0.68	[0.569, 0.790]	supported
Chronos small, detector survivors	0.98	[0.944, 1.000]	supported	0.70	[0.600, 0.800]	supported
Chronos small, 0.06σ	0.92	[0.891, 0.956]	supported	0.61	[0.539, 0.673]	supported
Chronos small, 0.03σ	0.87	[0.827, 0.909]	invalid (key)	0.62	[0.554, 0.682]	invalid (key)

Note. W is the win rate; parentheses name failed checks. The one sided randomization value was below 0.001 for every defined paired score and every single model score except frozen Moirai small ($p = 0.286$).

Table 10: Attribution gaps and diagnostic checks by model and setup.

Attribution gaps model and setup	J	paired G	single G
Main model comparisons (pooled ETTm1 and weather)			
Moirai small	102	$+0.2135$	$+0.1997$
Moirai base	95	$+0.2622$	$+0.3157$
Chronos small	97	$+0.1129$	$+0.1249$
Chronos base	99	$+0.2032$	$+0.1950$
Chronos large	99	$+0.2308$	$+0.2547$
Chronos-Bolt base	98	$+1.4751$	$+1.2423$
Toto base	93	$+0.3858$	$+0.3794$
Lag-Llama	101	$+8.7306$	$+8.4096$
Moirai-MoE small	98	$+0.2929$	$+0.2867$
Other host populations			
Chronos small, ETTm2	67	$+0.1703$	$+0.1745$
Chronos small, MetroPT-3	223	$+0.1818$	$+0.1920$
Chronos base, ETTm2	68	$+0.2280$	$+0.2274$
Chronos base, MetroPT-3	229	$+0.2645$	$+0.2751$
Toto base, MetroPT-3	232	$+0.3678$	$+0.3632$
Additional design checks (pooled ETTm1 and weather)			
Frozen Moirai small	94	n/a	$+0.0137$
Unfrozen Moirai small	104	n/a	$+0.2484$
Chronos small, 25% share	50	$+0.2101$	$+0.1989$
Chronos small, survivors	66	$+0.1969$	$+0.1917$
Chronos small, 0.06σ	99	$+0.0812$	$+0.0676$
Chronos small, 0.03σ	93	$+0.0639$	$+0.0677$

Diagnostic check values model and setup	paired key	single key	base neutrality	clean neutrality
Main model comparisons (pooled ETTm1 and weather)				
Moirai small	0.135	0.217	0.622	0.996
Moirai base	0.860	0.515	0.797	**0.048**
Chronos small	0.363	0.423	0.147	0.695
Chronos base	0.103	**0.024**	0.180	0.573
Chronos large	0.729	0.760	**0.062**	0.115
Chronos-Bolt base	**0.057**	0.135	0.506	0.615
Toto base	0.954	0.909	0.455	0.858
Lag-Llama	0.427	0.976	0.634	**0.085**
Moirai-MoE small	0.211	0.111	0.565	0.277
Other host populations				
Chronos small, ETTm2	0.703	0.683	0.307	0.312
Chronos small, MetroPT-3	0.925	0.982	0.337	0.720
Chronos base, ETTm2	0.466	0.564	0.445	0.680
Chronos base, MetroPT-3	0.678	0.547	0.493	0.690
Toto base, MetroPT-3	0.984	0.299	0.502	0.582
Additional design checks (pooled ETTm1 and weather)				
Frozen Moirai small	0.949	n/a	0.205	0.223
Unfrozen Moirai small	0.519	n/a	0.367	**0.057**
Chronos small, 25% share	0.549	0.624	0.779	0.144
Chronos small, survivors	0.188	0.204	0.145	0.385
Chronos small, 0.06σ	0.364	0.413	0.970	0.147
Chronos small, 0.03σ	**0.046**	**0.078**	0.146	0.465

Both panels of Table 10 use the same rows as Table 9; J counts physical clusters. Diagnostic values are probability values; bold values failed the 0.10 check. Paired key and single key are the two key isolation checks; n/a means the score was not defined. The quarter marked and detector survivor experiments reused the full share clean model. Attribution gap units differ for Lag-Llama, Moirai-MoE, and Chronos-Bolt, so gaps are compared only within a model family.

The mark visibility check met its reporting threshold in every experiment and did not affect the decision after the initial frozen experiment. The signed Chronos-Bolt value was negative because eight unmarked windows had flat contexts at the instance normalizer's floor, as anticipated before the run.

Only key isolation can invalidate the paired model score. Under a two sided threshold of 0.10, this check has a 10% chance of incorrectly rejecting a valid result. The single model score also requires clean training neutrality. If the checks are independent, their combined chance of incorrectly rejecting a valid single model result is about 19%. The initial frozen experiment used the original rule and is excluded. Of the nineteen later experiments governed by clean training neutrality, seventeen trained fresh clean models; the quarter marked and detector survivor experiments reused the full share clean model. Three of the seventeen fresh models failed the check. Under a binomial failure probability of 0.10, the probability of at least three failures in seventeen is approximately 0.238. The observed count is therefore not unusual under the stated check behavior.

F.2 How the final settings were selected

Figure 5 plots the Table 11 data. For each encoder, the first four rows vary training duration at 0.12σ; the rest vary amplitude at 3000 steps. The shared 0.12σ, 3000 step setting appears once. A dagger marks nine cells completed after selection.

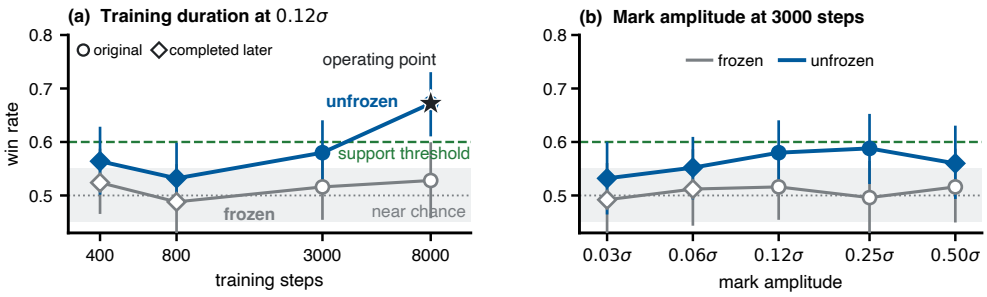

Figure 5: Study of the selected settings. (a) Win rate at amplitude 0.12σ as training duration increases. (b) Win rate at 3000 steps as mark amplitude increases. Error bars show reported 95% cluster bootstrap intervals. Horizontal guides mark $W = 0.60$ and chance, and the star marks the selected setting. The marker key distinguishes the original cells from the nine cells completed later. Both horizontal axes are logarithmic.

When the encoder weights remain fixed, the result changes little as either amplitude or training duration increases. From 400 to 8000 steps at 0.12σ and from 0.03σ to 0.50σ at 3000 steps, the win rate stays between 0.488 and 0.528, every interval covering chance. Allowing the encoder weights to update produces a larger response at the same amplitude. At 400 and 800 steps, the win rates are 0.564 and 0.532, both below the 3000 step result. At 3000 steps the interval is above chance (0.580), and the win rate is larger at 8000 steps (0.672). Amplitude matters only at the bottom of the range: at 0.03σ the frozen and unfrozen intervals both include chance, at 0.06σ it reads 0.552, and from 0.12σ to 0.50σ the unfrozen response is flat (0.580, 0.588, 0.560). These selection data show that allowing the model weights to change matters more than increasing the amplitude. The later studies therefore retain the smaller 0.12σ mark and use the unfrozen, 8000 step setting. The nine cells marked † in Table 11 were run on 2026-09-20, after the main experiment sequence, to complete the grid. Their results did not change the selected setting or decision rule.

Table 11: Study of the selected settings. The frozen 0.12σ, 3000-step row repeats the five-seed initial experiment in Tables 9 and 10; the other rows are single-run checks and do not add to the main experiment count.

Encoder	Amplitude	Steps	G	p_{rand}	W [95% CI]
frozen†	0.12σ	400	$+0.0228$	0.184	0.524 [0.468, 0.585]
frozen†	0.12σ	800	$+0.0261$	0.147	0.488 [0.427, 0.556]
frozen	0.12σ	3000	$+0.0137$	0.286	0.516 [0.457, 0.577]
frozen	0.12σ	8000	$+0.0067$	0.403	0.528 [0.460, 0.597]
frozen†	0.03σ	3000	$+0.0004$	0.491	0.492 [0.428, 0.558]
frozen†	0.06σ	3000	$+0.0054$	0.397	0.512 [0.446, 0.584]
frozen	0.25σ	3000	$+0.0089$	0.338	0.496 [0.431, 0.562]
frozen	0.50σ	3000	$+0.0117$	0.283	0.516 [0.452, 0.585]
unfrozen†	0.12σ	400	$+0.0444$	0.070	0.564 [0.504, 0.626]
unfrozen†	0.12σ	800	$+0.0336$	0.141	0.532 [0.474, 0.596]
unfrozen	0.12σ	3000	$+0.1021$	0.0003	0.580 [0.527, 0.638]
unfrozen	0.12σ	8000	$+0.2210$	$< 10^{-4}$	0.672 [0.613, 0.728]
unfrozen†	0.03σ	3000	$+0.0240$	0.078	0.532 [0.467, 0.598]
unfrozen†	0.06σ	3000	$+0.0457$	0.013	0.552 [0.495, 0.607]
unfrozen	0.25σ	3000	$+0.0988$	$< 10^{-4}$	0.588 [0.524, 0.650]
unfrozen†	0.50σ	3000	$+0.0830$	0.001	0.560 [0.496, 0.628]

F.3 Protocol evolution and deviations

The first experiment used a frozen encoder and returned a null. Its mark visibility check was positive, revealing that the original rule incorrectly required a learnable mark to look invisible to the base model. The second study reported visibility as a measure and tested the single model score at the new operating point. Its clean training neutrality check failed, showing that an unlucky key assignment could invalidate the single model contrast. The third study introduced the paired model score and retained the single model score as a secondary result. All subsequent studies use the paired and single model scores with their separate required checks. The two experiments after the sweep also redrew every random choice used for selection.

The clean training neutrality check also failed for the single model result in the paired Moirai base experiment. In both that experiment and the earlier unfrozen single model Moirai experiment, the clean model gap had the same sign across the two key ranges and was concentrated in weather. This pattern is consistent with an unlucky realized key difference rather than a systematic asymmetry between the key ranges. The finding does not make either single model result valid. It illustrates why the matched role swap is needed.

A preliminary Chronos execution used unintended synthetic background data because a required launch setting was omitted. An automated setup check identified the mismatch. That execution is excluded from all results tables and pooled summaries. Its paired model win rate was 0.952 $[0.925, 0.975]$, and its result points in the same direction as the included execution. It is retained only as an exploratory check. The reported Chronos result does not depend on it.

The Toto study required 128 input samples instead of 96 and used a 192 sample scoring window. Its host layout, cluster definition, and score definition remained unchanged. Toto training crops overlapped scoring windows more strongly than the Chronos crops, with mean overlap of 58.2% rather than 33.2%. Consequently, its raw gap and single model win rate are not evidence of a stronger mechanism across model families. Win rate remains the only comparison across families used in the paper. A later check found that the checkpoint's scale limit was not reached by the scoring inputs or any possible training crops.

F.4 How many hosts must be marked

The 25% experiment marked 62 hosts in 50 physical clusters. Its paired model win rate was 0.919 $[0.850, 0.982]$, and its single model win rate was 0.677 $[0.569, 0.790]$. Every applicable diagnostic check passed. The additional 50% study produced a paired model win rate of 0.960 $[0.923, 0.991]$.

At 10%, the point estimate remained 0.920 $[0.800, 1.000]$, but key isolation failed at $p = 0.068$, so this point cannot support attribution.

The paired model gap increases as the marked share falls. Recomputing the full share result on the same hosts accounts for only about 16% to 24% of that increase. The remaining difference is not explained by this host comparison. The data therefore support attribution at one quarter marking, but they do not support a claim that lower marking shares strengthen the mechanism.

G Robustness and practical behavior

This appendix asks whether the mark raises forecast error, whether attribution survives when marked training spans are changed, and whether a party that knows the scheme but not the key can detect or reduce the signal. These exploratory studies do not expand the scope of the main results.

G.1 Forecast utility and transform robustness

At the fixed amplitude of 0.12σ, ridge forecasting on ETTm1 changed MAE by $+0.2\%$ with a 95% interval of $[-0.3, 1.0]\%$; seasonal naive forecasting on ETTm1 changed it by $+3.3\%$ $[0.4, 7.2]\%$. On weather, the changes were $+9.2\%$ $[2.9, 16.7]\%$ for ridge and $+0.5\%$ $[0.2, 0.8]\%$ for seasonal naive. The ETTm1 ridge interval includes zero; the other three intervals do not. At 0.25σ or above, all four checked combinations had positive intervals, so the selected amplitude does not guarantee unchanged forecast error.

The training-data transformation study used one training run and 102 clusters with Moirai small. Each model trained on transformed marked spans and was scored on the owner's unchanged keyed windows. Without a transformation, the single model win rate was 0.636. Eight bit quantization of the spans retained support at 0.680 $[0.632, 0.727]$, as did noise at half the mark's standard deviation at 0.608 $[0.545, 0.668]$. Full scale noise, per window standardization, and downsampling by two failed the full support rule. Downsampling changes the keyed frequency grid and shows a tested limit after transformation.

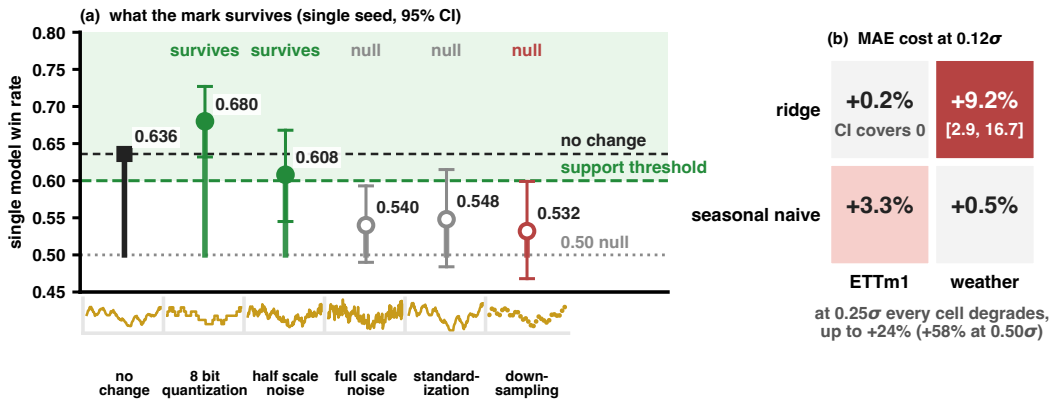

Figure 6: Robustness of model attribution and forecast cost. Panel (a) reports single model win rates after each marked training span transform. Panel (b) reports MAE changes at 0.12σ by dataset and forecaster, with the higher amplitudes noted.

G.2 Detection and removal without the key

Unlike the mark visibility check in §5.4, this study asks whether a party without the key can find the mark. It used the 250 marked spans from the full share Chronos experiment in 97 physical clusters. The simulated outside party knew the scheme and had the released spans, but not the key or the owner's clean series. It trained two classifiers on marked and unmarked simulations from ETTm2, ETTh1, ETTh2, and Exchange rate, all outside the owner's pool. Two direct peak statistics required no training.

The two peak statistics have no classification threshold, so TPR and FPR do not apply. The last two rows of Table 12 repeat the probe on the same population at smaller amplitudes and report the detector selected using simulated validation data.

Table 12: Detection of the full share Chronos mark without the key. Classifier thresholds keep the false positive rate at or below 5%; intervals use a physical cluster bootstrap.

Detector	AUROC	ETTm1 AUROC	Weather AUROC	TPR	FPR
Three spectral peaks	0.910 [0.882, 0.936]	0.838	0.924	n/a	n/a
Referenced peak score	0.865 [0.832, 0.894]	0.772	0.913	n/a	n/a
Logistic regression	0.944 [0.921, 0.964]	0.852	0.970	0.608 [0.541, 0.665]	0.004 [0.000, 0.012]
Boosted trees	0.949 [0.928, 0.967]	0.848	0.979	0.648 [0.580, 0.705]	0.024 [0.008, 0.044]
Boosted trees at 0.06σ	0.889 [0.857, 0.916]	0.675	0.951	0.428 [0.359, 0.496]	0.008 [0.000, 0.020]
Logistic regression at 0.03σ	0.776 [0.735, 0.813]	0.555	0.848	0.308 [0.240, 0.379]	0.008 [0.000, 0.020]

The boosted tree detector identified 64.8% of marked spans at a 2.4% false positive rate. Dropping them left 88 of 250 spans in 66 of 97 clusters. Because this subset was selected, the 25% result does not apply. A separate experiment trained on the 88 surviving marked spans, with clean spans replacing those dropped. Its single model score was supported. Its paired score was also supported, but its matched model trained on sham marks that the detector had not screened, so that score is an upper bound for this removal setting (Table 9).

Three removal procedures were applied to every marked span. A three peak notch retained a median 61.7% of the mark's RMS amplitude; least squares subtraction at three or five peaks retained 52.5% and 56.6%. The best procedure reduced the mark to at most one quarter on 44% of hosts while adding 0.044σ distortion to an unmarked span. These peak-removal tests show reduction, not attribution after removal.

G.3 SIGNAL SURVIVAL AFTER DATA TRANSFORMATIONS

The matched filter was evaluated before and after seven data transformations on real ETTm1 and weather windows, with a separate simulated null for each setting. This tests one window, not trained model attribution.

Table 13: Signal survival after data transformations. Each row uses 1000 marked and 1000 null trials (Monte Carlo standard error about 1.6 percentage points). Source lengths and simulated null thresholds vary, so detection power is not directly comparable across rows.

Transformation	Source samples	Mean score	Detection power
No transformation	128	3.131	0.636
Downsampling by two	256	3.393	0.695
Downsampling by four	512	2.758	0.437
Moving average over three samples	128	3.043	0.527
Moving average over seven samples	128	2.877	0.506
Noise at half the mark scale	128	2.937	0.572
Noise at the full mark scale	128	2.689	0.603
Off grid resampling by 2.37	302	2.597	0.409

Smoothing and off grid resampling had lower mean scores and detection power than the unchanged setting in this probe.

The signal level test asks whether spectral energy remains in one transformed window. The model level test asks whether a model trained after resampling still prefers the original keyed window. Signal survival is needed for attribution after a transformation, but it is not enough.