TRACS: TIME-SERIES REBALANCING AND COVERAGE-AWARE SELECTION FOR PRETRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

As time-series foundation models (TSFMs) mature, performance increasingly depends on the quality and composition of their pretraining data. Most existing data selection methods rely on training feedback or external supervision, which makes it difficult to determine the selected corpus entirely before pretraining. We present **TRACS** (**T**ime-Series **R**ebalancing **a**nd **C**overage-Aware **S**election), a zero-feedback method for selecting time-series pretraining data without either source of supervision. TRACS uses GMM-based soft clustering to organize candidate series by temporal pattern, then assigns sampling quotas based on the coverage required within each cluster rather than on cluster size. The selected subset and its pattern coverage can therefore be determined before pretraining begins. Under the stated assumptions, we prove coverage of all reachable patterns, finite-step termination of the missing-pattern repair procedure, and a bounded representation error with respect to the rebalanced target distribution. We evaluate TRACS on MOMENT-1 Small, PatchTST, and Timer under the same 10B-token pretraining budget. Using only 25% or 50% of the candidate corpus, TRACS attains the lowest aggregate MASE and CRPS among the evaluated pretraining baselines on 20 stable zero-shot forecasting tasks from GIFT-Eval. Relative to pretraining on the full corpus under the same token budget, TRACS reduces MASE by 2.2%–4.0% and provides an explicit guarantee on pattern coverage that the baseline methods do not offer.

1 INTRODUCTION

Time-series foundation models (TSFMs) underwent rapid architectural iteration between 2023 and 2025. Chronos (Ansari et al., 2024) first established the pretraining template by training a T5 model on tokenized time series. Moirai (Woo et al., 2024) then introduced multi-patch-size encoding. TimesFM (Das et al., 2024) showed that a 200M-parameter decoder-only architecture matched specialized models on zero-shot forecasting. Most recently, TimeMoE (Shi et al., 2025) reported state-of-the-art results through a sparse mixture-of-experts design. These works collectively validated the effectiveness of large-scale pretraining. However, as architectures converge, the performance bottleneck is shifting from architectural design to the quality and composition of pretraining data, because pretraining corpora have grown to hundreds of billions of observations, far exceeding what model training can consume.

The growth of pretraining corpora substantially increases computational cost. Taking LOTSA (Woo et al., 2024) as an example, this corpus covers about 27.6 billion observations, and a single full training run costs up to several thousand GPU-hours. Data in domains such as the Internet of Things, industrial sensing, and financial trading continues to grow rapidly. Yet blindly expanding the corpus not only yields diminishing marginal returns but also introduces systematic bias. Long sequences are over-represented in sliding-window construction, high-frequency patterns suppress low-frequency signals, and specific deployment scenarios exhibit unpredictable failures. Consequently, precisely identifying high-value subsets from massive corpora before pretraining, rather than passively receiving the full data, is a prerequisite for reducing computational overhead and improving model robustness (Figure 1). A growing number of studies have begun to address this challenge. BLAST (Shao et al., 2025), Training Diet (Pennino & Gabbrielli, 2026), and LP-Mirror (Ao & Chen, 2026) explore data selection from the perspectives of balanced sampling, weight optimization, and influence-score evaluation, respectively. Among them, BLAST is a pretraining-time fixed-feature method, whereas

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.

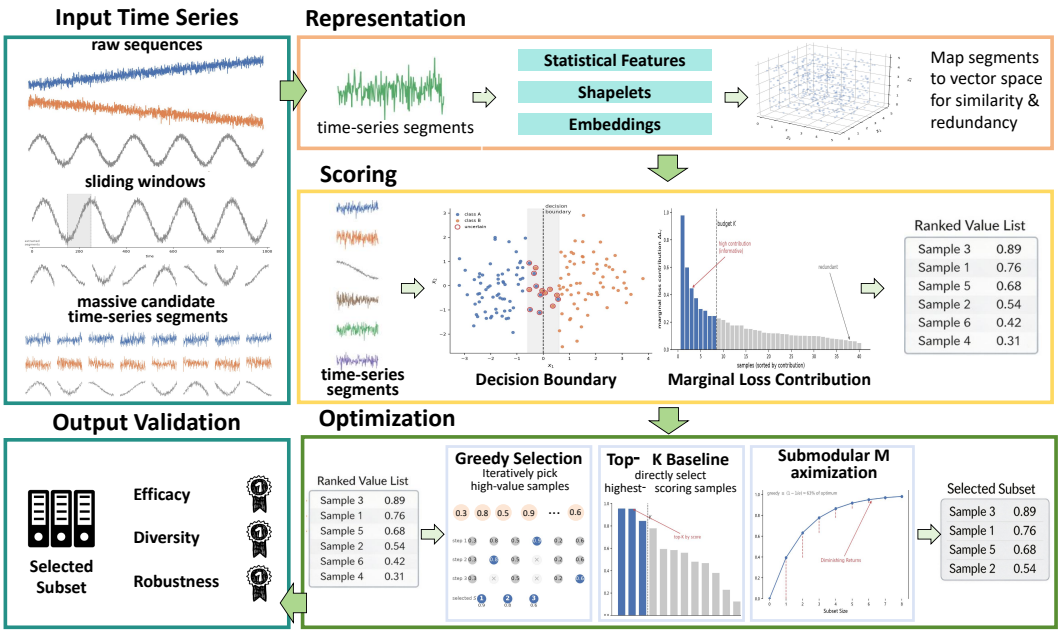


Figure 1: Schematic of time-series data selection for pretraining, from the full candidate corpus to a compact subset selected under diversity and coverage constraints before pretraining.

Training Diet and LP-Mirror rely on selection-time model feedback and guidance-set signals, respectively. This paper further asks how data should be selected by time-series structure before pretraining, without querying the model to be trained or using validation- or guidance-set feedback.

To address this problem, we propose TRACS (Time-Series Rebalancing and Coverage-Aware Selection), which performs pretraining-time data selection without model feedback by maximizing diversity under coverage constraints, reducing continuous coverage holes under a fixed budget, and enforcing reachable-pattern coverage. This is difficult for four reasons.

(1) **The search space is enormous.** The corpus yields too many subsets to consider one by one, and selection itself has to cost less than the training it is meant to save, or it defeats its own purpose.

(2) **Temporal structure has no ready-made description.** Selecting data by temporal structure first requires a way to describe a series' temporal behavior. NLP has off-the-shelf embeddings such as SBERT that measure redundancy directly, but time series have no equivalent. A selector must build such a description itself, and it has to be cheap to compute before pretraining, able to tell behaviors apart, and still verifiable.

(3) **Diversity does not guarantee coverage.** Space-filling methods (Niederreiter, 1992) maximize diversity and spread in a continuous feature space, but what matters is that every reachable pattern must be kept, which is a discrete, combinatorial condition. Maximizing a continuous objective does not by itself satisfy it.

(4) **Without feedback, the result cannot be verified.** Selection runs with no feedback at any stage, meaning it never queries the model to be trained, uses a validation set, or adjusts according to selection outcomes. This is distinct from never using downstream signals at all, and the three-layer division is detailed in Appendix A. Because of this, a chosen subset cannot be judged before pretraining, so a selector must optimize a proxy that predicts downstream quality and must confirm, from the selected data alone, that the proxy's conditions are met.

These four challenges shape the design of TRACS, from problem definition through algorithm implementation to experimental validation; each design choice is motivated by one of them. The contributions of this paper are as follows.

(1) We characterize zero-feedback pretraining-time time-series data selection as an optimization problem under coverage constraints, decoupling into K independently solvable within-cluster sub-

Table 1: Comparison of the design orientations of data selection methods. The six axes (left to right) are downstream-signal utilization, target-domain customization, spatial-organization granularity, diversity-mechanism explicitness, selection-phase compute, and representation model-agnosticism. Stars indicate each method's strength per axis (distinguishing design orientation, not quality).

Method	Signal	Target	Granularity	Diversity	Compute	Model-agnostic
BLAST	⋆	⋆	⋆⋆	⋆⋆	⋆	⋆⋆⋆
Training Diet	⋆⋆⋆	⋆⋆	⋆	⋆	⋆⋆⋆	⋆
LP-Mirror	⋆⋆	⋆⋆⋆	⋆	⋆	⋆⋆	⋆
TRACS (ours)	⋆	⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆	⋆⋆⋆

problems, and give a computationally tractable five-stage implementation that organizes heuristic sampling operations into a verifiable constrained-selection workflow.

(2) We construct seven-dimensional structured and 20-dimensional extended statistical features via STL decomposition and generic statistics, yielding a 27-dimensional model-agnostic representation that requires no pretrained encoder and grounds clustering and coverage on interpretable features.

(3) Beyond statistical balancing, we establish conditional, verifiable results for coverage and representativeness, namely a sufficient condition connecting continuous filling to six-bit coverage, finite seven-bit repair under feasibility,

and Wasserstein and Lipschitz bounds relative to the rebalanced target distribution.

(4) We empirically evaluate TRACS across three heterogeneous architectures (MOMENT-1 Small, PatchTST, and Timer). Under a fixed 10B-token budget, using only 25% or 50% of the candidate corpus, TRACS attains the lowest aggregate MASE and CRPS among all compared methods and outperforms full-corpus pretraining on both metrics across the three architectures.

2 Related Work

BLAST (Shao et al., 2025) encodes seven types of fixed statistical indicators into one-hot features, projects them into a two-dimensional UMAP embedding, and partitions the space into a 100×100 grid. It then samples non-empty grid cells with equal probability, selects series and windows within each cell, and applies grid mixup, with all rules fixed before pretraining. Training Diet (Pennino & Gabbrielli, 2026) embeds the corpus with a pretrained encoder, clusters it into 36 groups via K-Means, and searches the cluster sampling weights with Optuna according to the validation MSE, so its selection rule changes with model performance. LP-Mirror (Ao & Chen, 2026) trains a channel-independent linear proxy on the source data, adapts it to a small guidance set with one gradient step, and scores each candidate by the resulting loss change, which aligns the candidate gradient with the guidance-set gradient and injects target-domain signal. TRACS and BLAST are both pretraining-time, feature-driven methods, but TRACS is model-agnostic and treats structural coverage as both the selection objective and the output constraint rather than relying on downstream performance feedback, as summarized in the comparison table below. The BLAST-style baseline here adapts the original method (see Section 5.1 and Appendix A.1), so the comparison targets the two framework implementations, and the problem-formulation comparison is in Appendix B.3. For representation, TRACS does not rely on a pretrained encoder but constructs TIME-inspired features (Qiao et al., 2026) from STL decomposition and hand-crafted statistics, whose full genealogy and definitions are in Appendix A.2 and Appendix C. The problem definition is in Section 3, and the selection-objective rationale and alternatives are in Appendix B.

3 Problem Formulation

This section formalizes the above design as a subset selection problem under coverage constraints. Given a candidate corpus and a target size m, TRACS needs to improve continuous coverage in the six-dimensional space and ensure that the final subset covers all reachable seven-bit patterns. The complete notation and core symbols are as follows.

Notation. Let $D = \{x_1, \ldots, x_n\}$ be a corpus of n univariate time series, each serving as one selection unit. After Stage 0–1 feature extraction, each x_i maps to $f_i \in \mathbb{R}^{27}$, whose first 7 dimensions $f_i^{1:7} = (F_{i1}, \ldots, F_{i7})$ are the core structured features and whose last 20 are extended statistical features. TRACS selects a subset D' of size $m < n$ improving continuous coverage in the six-dimensional space of $F_{1:6}$. The seven-bit reachable patterns are checked separately by Stage 4.

Under this setting, TRACS decomposes diversity into two levels. First, GMM-EM produces soft posterior assignments, while the hard labels partition the corpus into K clusters $C_1, \ldots, C_K$. For each x_i, GMM outputs a posterior probability vector $p_i = (p_{i1}, \ldots, p_{iK})$, where $p_{ik} = P(k \mid f_i)$ and $\sum_k p_{ik} = 1$. The hard label $y_i = \arg\max_k p_{ik}$ is used for quota counting. The posterior probabilities are used to characterize the cluster structure and to perform entropy-guided sampling of narrow clusters (Section 4.4). At the inter-cluster level, the subset is required to cover all K clusters, with an initial rebalanced quota per cluster given by $q_k^{(0)} \in \{\lfloor m/K \rfloor, \lceil m/K \rceil\}$ and $\sum_k q_k^{(0)} = m$. After any capacity shortfall is transferred to clusters with spare capacity, let $q_k = |D' \cap C_k|$ be the realized Stage 3 count. Then $0 \le q_k \le n_k$ and $\sum_k q_k = m$, although q_k need not remain in the two-value set above. This construction rebalances the fixed budget rather than reproducing the empirical distribution. Within-cluster coverage is measured by the filling distance $d_H(S_k) = \sup_{x \in \Omega_k} \min_{i \in S_k} \|f_{i,1:6} - x\|_2$, where $\Omega_k = \prod_{j=1}^{6} [\ell_k^{(j)}, u_k^{(j)}]$ is the bounding hyper-rectangle of cluster k over F_1–F_6, with $\ell_k^{(j)} = \min_{i \in C_k} F_{ij}$ and $u_k^{(j)} = \max_{i \in C_k} F_{ij}$. $d_H(S_k)$ is the largest empty-ball radius in Ω_k (the covering radius, not the Hausdorff distance). The smaller this value, the smaller the largest coverage hole. The filling distance is compared with alternative objectives in Appendix B.2. The within-cluster selection is implemented in Section 4.4 and Appendix C.4. Under realized quotas q_k, the formal filling objective decomposes into K local subproblems. The seven-dimensional retrieval is linked to this objective by the projected bound in Theorem 1.

Although continuous filling controls the geometric covering radius in the six-dimensional space, it cannot directly guarantee discrete pattern completeness. Rare patterns near the median split thresholds may still be missed. To check these omissions, TRACS uses the global median of F_1–F_6 as the threshold and takes the original binary value of F_7 directly (the stationarity indicator $I(p_{\mathrm{ADF}} < 0.05)$; see Appendix C) to define the pattern code. It sets $b_{ij} = \mathbb{I}[F_{ij} > \widetilde{F}_j]$ for $1 \le j \le 6$, keeps $b_{i7} = F_{i7}$, and encodes the seven-bit pattern $\mathrm{pid}(i) = \sum_{j=0}^{6} b_{i,j+1} \cdot 2^j \in \{0, \ldots, 127\}$. Full coverage requires at least one sample for every code reachable in the corpus (102 in the LOTSA subpool), and this representativeness metric is evaluated separately from downstream performance.

Before the seven-bit coverage verification, the continuous selection problem of Stage 3 minimizes $\max_{k=1}^{K} d_H(D' \cap C_k)$ over subsets $D' \subset D$ of size m, subject to $|D' \cap C_k| = q_k$, where the q_k are the realized, capacity-corrected quotas. Stage 4 uses this set as the starting point to verify the seven-bit patterns. A cross-cluster Swap keeps $|D_{\mathrm{final}}| = m$ and full pattern coverage, at the cost of deviating from q_k by at most the number of replacements.

4 METHOD

Section 3 defined the inter-cluster quotas, the six-dimensional filling objective, and the seven-bit pattern coverage requirement. TRACS implements this formulation in five stages (Figure 2), none of which uses model feedback. Stage 0 diagnoses STL applicability. Stage 1 extracts the seven-dimensional structured features and 20-dimensional extended features. Stage 2 performs incremental PCA and coreset GMM-EM. Stage 3 allocates rebalanced quotas with capacity correction, followed by seven-dimensional Sobol QMC retrieval or entropy-guided selection; its formal continuous-coverage analysis is defined on the F_1–F_6 projection. Stage 4 checks the seven-bit codes and performs a Swap when a pattern is missing.

4.1 STAGE 0: STL APPLICABILITY DIAGNOSIS

Stage 0 assigns each sequence to one of four paths: STL, STL_NOSSN, FALLBACK, and REJECT, according to its applicability to STL decomposition (Algorithm 1, Appendix C). The additive assumption of STL ($x = T + S + R$) may fail when a sequence is too short, has no seasonality, or is irregularly sampled, and forcing decomposition adds overhead. Stage 0 routes each sequence through four empirical-threshold checks: missing rate, length/variance, sampling-interval regularity,

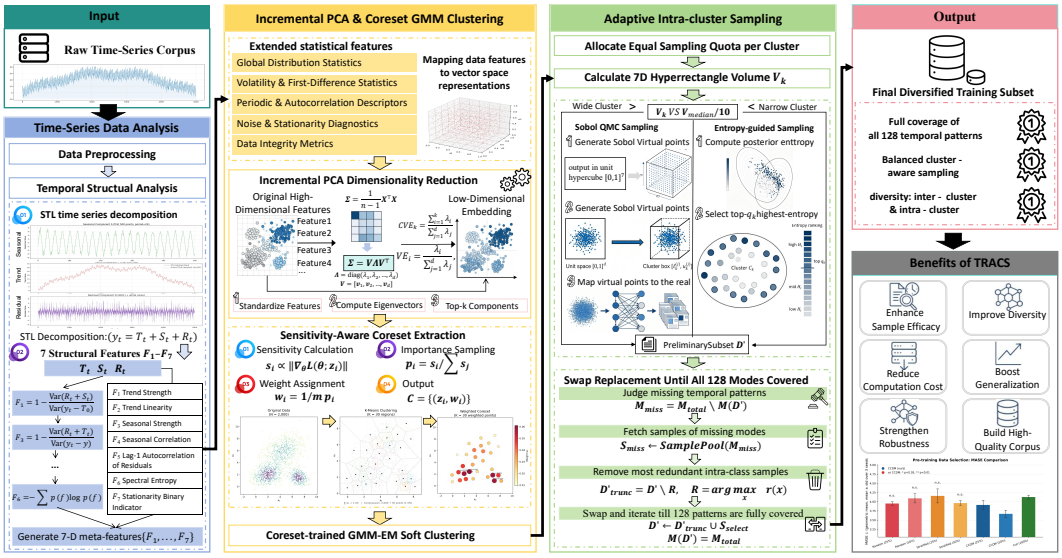


Figure 2: The five-stage TRACS pipeline, from STL diagnosis and feature extraction through coreset clustering and coverage-aware quota allocation to seven-bit pattern verification and repair. Of the 128 theoretical codes, 102 are reachable in the LOTSA subpool.

and first-difference ACF periodicity detection. Even when a period is detected but the sequence is too short for two periods ($L < 2 \times P$), the trend-residual split is retained, so residual autocorrelation (F_5) and spectral complexity (F_6) remain uncontaminated by the trend component.

4.2 Stage 1: Feature Extraction

After Stage 0 completes the routing, Stage 1 constructs fixed-length hand-crafted features for each sequence. This paper uses seven-dimensional TIME-inspired structured features F_1–F_7 and 20-dimensional extended statistical features, concatenated as the 27-dimensional GMM input. F_1–F_6 are also used by Stage 3 for continuous filling, and the complete F_1–F_7 are used by Stage 4 for encoding. The concrete definitions and differences from the TIME formulas are in Appendix C.

4.3 Stage 2: Gaussian-Mixture Clustering

To address cross-feature correlations (which a diagonal-covariance GMM does not model) and the excessive EM cost at the scale of $n > 10^7$, Stage 2 first applies incremental PCA to the z-score standardized features (retaining 95% variance, $d' \approx 15$–20), making the diagonal-covariance approximation more plausible and reducing the parameter count from $O(K \cdot d'^2)$ to $O(K \cdot d')$. It then uses a lightweight GMM to compute a heuristic proxy for sampling sensitivity, constructs a weighted subsample of $n_c = 10^5$, and runs weighted EM (Appendix C.2). Since this maximum-posterior proxy is not a formal sensitivity bound, the coreset cannot inherit Bachem's $O(1/\sqrt{n_c})$ guarantee and is an empirical heuristic (Appendix C.3).

4.4 Stage 3: Coverage-Aware Quota

Stage 3 assigns the base quotas and computes the seven-dimensional routing volume $V_k = \prod_{j=1}^{7} \max\{u_k^{(j)} - \ell_k^{(j)}, 10^{-15}\}$. Wide clusters with sufficient capacity generate $q_k^{(0)}$ scrambled Sobol targets in their $F_{1:7}$ bounding boxes and retrieve unrepeated samples through L2-normalized FAISS HNSW search. Narrow clusters use normalized GMM posterior entropy. Capacity-limited clusters contribute all available samples, and redistributed shortfalls yield the realized counts q_k. The binary F_7 is included as a retrieval heuristic so that stationarity can influence matching. No seven-dimensional continuous-coverage guarantee is claimed. The formal filling objective remains six-dimensional. Appendix D projects the targets onto $F_{1:6}$ and absorbs the seven-dimensional retrieval

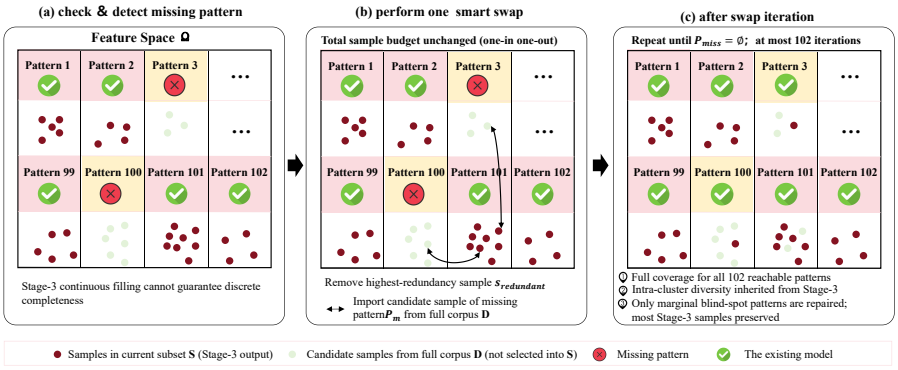


Figure 3: Stage 4 pattern check and repair. Every seven-bit reachable pattern is verified, and a missing one is restored by swapping in a candidate sample and removing a locally redundant one.

discrepancy into r_k (Equation (10)), thereby linking the implementation to Theorem 1. Stage 4 separately checks and repairs reachable seven-bit coverage (Algorithm 2 is in Appendix C.4).

Theorem 1 (Sufficient condition for six-bit reachable-combination coverage)**.** *Let $\bar{h}_k$ be an upper bound on the continuous filling radius of cluster C_k, Δ_{gap} the coverage gap of each cluster's bounding box relative to the global six-dimensional box, and $\lambda'_{\min}(D)$ the minimum side length of a six-bit reachable pattern cell. If $\lambda'_{\min}(D) > 0$ and $\max_{1\le k\le K} \bar{h}_k + \Delta_{\text{gap}} < \lambda'_{\min}(D)/2$, then the Stage 3 set S_0 covers every six-bit reachable combination, i.e., $\{c_6(i) : i \in S_0\} = P_6$ (assuming each cluster is non-empty, $m \ge K$). The complete notation, construction, and proof are in Appendix D.*

4.5 Stage 4: Pattern Repair

Stage 3 performs seven-dimensional retrieval but is analyzed through the continuous filling of F_1–F_6. The seven-bit encoding also contains the binary F_7, and Theorem 1 is only a sufficient but not necessary condition for six-bit coverage. Stage 4 therefore checks all reachable seven-bit patterns and corrects only when omissions occur (Figure 3). It counts the seven-bit patterns of the Stage 3 subset, returns that subset unchanged if the missing set M_{missing} is empty, and otherwise, in each round, adds one missing-pattern sample from the candidate corpus and removes one locally redundant sample from the currently most populated pattern to maintain the size m. The added and removed samples may belong to different GMM clusters, so the cluster counts may deviate from the Stage 3 quotas serving as the initial allocation. The sample to remove is the one with the smallest average cosine distance to its $k = 5$ nearest neighbors within the same pattern, namely $\bar{d}(i) = \frac{1}{k}\sum_{j\in\mathcal{N}_k(i)}\big(1 - \cos_\text{sim}(\mathbf{f}_i^{1:7}, \mathbf{f}_j^{1:7})\big)$.

As long as the removed pattern still has other samples, a single replacement does not create new omissions, the number of missing patterns decreases monotonically, and the loop terminates in finite time when M_{missing} is empty. The final subset always maintains size m and covers all reachable seven-bit patterns. The termination property and the effects of the replacements on the quotas and covering radius are characterized by Theorem 2 (Algorithm 3 is in Appendix C.4).

Theorem 2 (Pattern repair, finite termination, and perturbation bound of Stage 4)**.** *Let $R = |P_7|$ be the number of seven-bit reachable patterns and s_0 the number of initially missing patterns. Under $m \le n$, a size-m subset covering all of P_7 exists if and only if $m \ge R$. In this case Stage 4 terminates after exactly $s = s_0$ Swaps, and the final subset S_f satisfies $|S_f| = m$ and $\{c_7(i) : i \in S_f\} = P_7$. The quota drift, covering-radius increment, and sample-distribution displacement caused by the repair are all bounded (bounds and proofs in Appendix D).*

Theorems 1 and 2 guarantee six-bit coverage and seven-bit pattern repair, but both remain at the coverage level and do not touch distributional representativeness. Theorem 3 reuses the premises of Theorem 2 ($m \le n$, $m \ge R = |P_7|$), requiring no new assumptions.

Theorem 3 (Stability of the final empirical measure and Lipschitz functionals)**.** *Under the conditions of Theorem 2, let ε_f be the sum of the per-cluster quantization error, the quota imbalance, and the transport cost of the Stage 4 Swap. Then the empirical measure μ_{S_f} of the final subset satisfies*

$W_1(\mu_q, \mu_{S_f}) \le \varepsilon_f$ relative to the rebalanced target empirical measure μ_q. Furthermore, for any Lipschitz functional φ, we have $\left|\int \varphi\, d\mu_q - \int \varphi\, d\mu_{S_f}\right| \le \|\varphi\|_{\mathrm{Lip}} \varepsilon_f$. The concrete expression of ε_f and the remaining bounds (including that relative to the full corpus μ_I) are in Appendix D.

5 EXPERIMENTS

We evaluate TRACS across three heterogeneous architectures on zero-shot forecasting. At matched subset sizes and under the same training protocol and token budget, we compare TRACS with Random, Stratified, and BLAST-style selection. We then compare TRACS-selected subsets with pretraining on the full corpus under the same token budget. Ablations isolate the effects of quota rebalancing, within-cluster retrieval, pattern repair, and cluster granularity.

5.1 EXPERIMENTAL SETUP

Pretraining data and downstream evaluation. From LOTSA, we construct a candidate subpool of approximately 600K sequences after applying the predefined length filters. Subset methods retain 25% or 50% of this subpool, treating each sequence as one selection unit. Downstream evaluation uses the 28 candidate zero-shot forecasting tasks from GIFT-Eval (Aksu et al., 2024).

Compared methods. We compare TRACS with Random, Stratified, and BLAST-style, and use Full Corpus as the full-data reference. BLAST-style replaces the original 61-dimensional discrete features with F_1–F_7 and removes grid mixup, while retaining the original UMAP projection and grid partitioning. Its results therefore refer only to the adapted implementation. Training Diet and LP-Mirror rely on model feedback and are outside the zero-feedback comparison. Baseline implementation details are in Appendix A.1 and Appendix B.3.

Training models and protocol. Three heterogeneous architectures (MOMENT-1 Small (Goswami et al., 2024), PatchTST (Nie et al., 2023), and Timer (Liu et al., 2024), with 19–38M parameters) are randomly initialized and trained from scratch with AdamW over 5 seeds under a fixed 10B-token budget. Smaller subsets are trained for more epochs so all methods process the same token count.

Evaluation metrics and statistical aggregation. Among the 28 candidate tasks, 7 have undefined MASE, and `covid_deaths` is excluded because its seasonal-naive denominator is near zero and its MASE reaches 10^8–10^9. Table 2 reports the geometric mean over the remaining 20 stable tasks, together with per-task Win/Tie/Loss and paired bootstrap results. The bootstrap resamples per-task differences 10,000 times. The 95% intervals quantify uncertainty across tasks.

We additionally report pattern coverage as the fraction of the 102 reachable seven-bit patterns represented in each selected subset. Experimental hyperparameters (Table 6), statistical conventions, and coverage definitions are provided in Appendix E, with per-task MASE and paired-bootstrap details in Appendix F.

5.2 MAIN RESULTS

Table 2 and Figure 4 summarize the main forecasting results. Across MOMENT, PatchTST, and Timer, TRACS attains the lowest aggregate MASE and CRPS, although the better retained fraction varies by architecture and metric. At matched subset sizes and under the same training protocol and token budget, TRACS consistently achieves lower aggregate errors than the adapted BLAST-style baseline. Even with only 25% of the candidate corpus, TRACS outperforms Full Corpus on both aggregate metrics across all three architectures. Using the better TRACS configuration for each architecture, the MASE reduction relative to Full Corpus ranges from 2.2% to 4.0%. Because all methods process the same 10B training tokens, the improvement cannot be attributed to a larger training budget, indicating that how the pretraining corpus is organized remains important even when the number of processed tokens is fixed. Increasing the retained fraction from 25% to 50% does not uniformly improve performance, so the preferred subset size depends on the forecasting architecture and evaluation metric. Per-task results and paired bootstrap analyses in Appendix F show that the gains vary across tasks and that the statistical evidence is not uniform across comparisons. The coverage analysis helps explain what distinguishes the selected subsets. As shown in Table 14, Random, Stratified, and BLAST-style retain all reachable stationary patterns but miss several non-stationary ones, whereas both TRACS subsets retain all 102 reachable seven-bit patterns.

Table 2: Main results report MASE geometric mean and CRPS (↓, mean ± std over 5 seeds). Bold and underline mark the best and second-best values per column.

	MASE ↓			CRPS ↓		
Method	**MOMENT**	**PatchTST**	**Timer**	**MOMENT**	**PatchTST**	**Timer**
Random-25	1.0974±0.0182	1.6663±0.0171	<u>1.4311±0.0243</u>	48.38±1.96	77.14±0.28	65.26±1.13
Random-50	1.0823±0.0133	1.6349±0.0151	1.4312±0.0172	49.14±0.36	75.67±0.96	65.41±0.39
Stratified-25	1.0902±0.0114	1.6547±0.0674	1.4883±0.0362	48.80±1.39	76.74±2.40	68.12±1.40
Stratified-50	1.0988±0.0129	1.6853±0.0303	1.4500±0.0279	49.51±0.34	77.70±1.33	66.09±1.34
BLAST-style-25	1.0897±0.0455	1.6685±0.0499	1.4796±0.0143	46.90±1.75	77.38±2.13	67.31±0.34
BLAST-style-50	1.0912±0.0088	1.6290±0.0216	1.4501±0.0132	47.78±0.40	75.87±0.74	66.09±0.72
TRACS-25	<u>1.0593±0.0177</u>	**1.5960±0.0369**	1.4331±0.0315	**46.26±0.57**	<u>74.86±1.77</u>	<u>65.00±1.45</u>
TRACS-50	**1.0500±0.0232**	<u>1.6060±0.0295</u>	**1.4091±0.0314**	<u>46.44±1.08</u>	**74.43±1.22**	**64.20±1.46**
Full Corpus	1.0843±0.0094	1.6314±0.0097	1.4674±0.0767	49.61±1.17	75.76±0.38	66.65±3.32

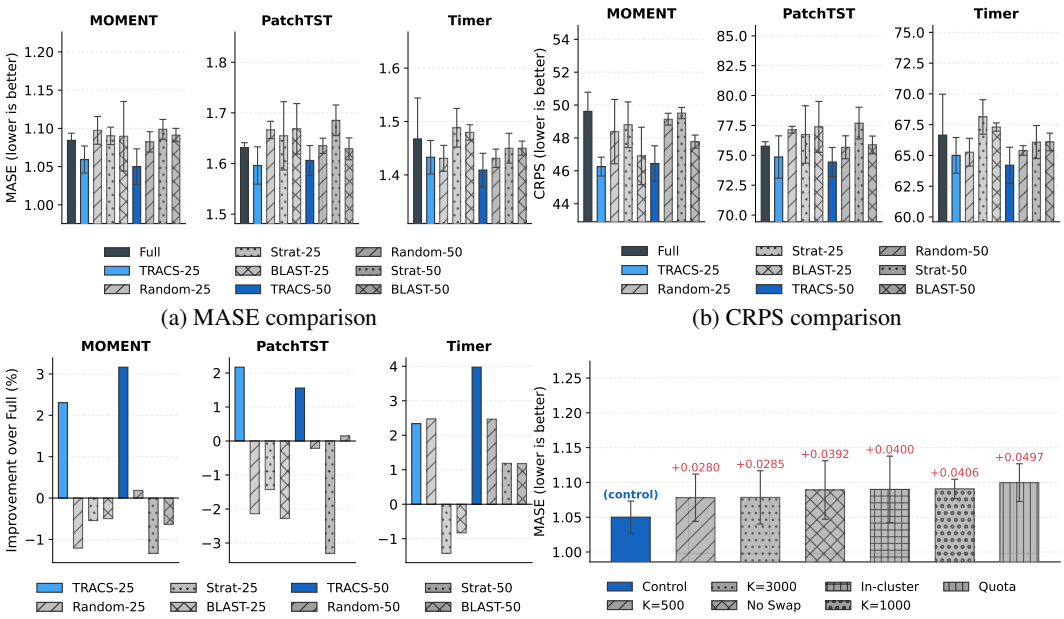
Table 3: Ablation results (MOMENT, 50% subset, 5 seeds, MASE geometric mean ↓).

Ablation variant	MASE	Δ vs. ref.	CRPS
TRACS (K=2000) [ref.]	1.0500 ± 0.0232	0	46.44
K=500	1.0780 ± 0.0340	+0.0280	47.97
K=1000	1.0906 ± 0.0140	+0.0406	47.54
K=3000	1.0785 ± 0.0383	+0.0285	48.08
In-cluster random (no Sobol)	1.0900 ± 0.0478	+0.0400	47.79
No Swap (Stage 3 only)	1.0892 ± 0.0420	+0.0392	48.17
Proportional quota (no rebalancing)	1.0997 ± 0.0271	+0.0497	50.23

These patterns capture differences in trend, seasonality, residual dependence, spectral structure, and stationarity, so their retention preserves temporal behaviors that can be lost during corpus reduction. Full Corpus also covers all 102 patterns but yields higher aggregate forecasting errors than TRACS, showing that complete pattern coverage alone is not sufficient. The allocation of the finite budget across behavioral clusters and the choice of samples within each cluster also matter. TRACS handles these two aspects through quota rebalancing and within-cluster filling, while the final repair step prevents reachable patterns from being omitted. This interpretation is examined in the ablations: proportional quotas and random within-cluster selection retain complete final coverage but perform worse than the full TRACS configuration, whereas removing the repair step leaves reachable patterns uncovered. Theorem 3 addresses the distributional aspect of the same construction by bounding, under its stated conditions, the Wasserstein distance between the final subset and the rebalanced target distribution in the projected feature space. The theorem therefore provides a guarantee on representational deviation, while the coverage, forecasting, and ablation results show how the selected corpus behaves in terms of structural completeness and downstream utility, allocation, retrieval, and repair.

5.3 Ablation Study

Tables 3 and 14 compare the main TRACS components under the same MOMENT subset size and training budget. The full configuration achieves the lowest aggregate MASE and CRPS. Proportional quota causes the largest degradation despite retaining complete reachable-pattern coverage. Since retrieval and repair are unchanged, the gap isolates budget allocation: proportional allocation preserves cluster frequency, allowing common behaviors to consume more of the fixed budget while rarer regions receive fewer representatives. This shows that corpus frequency is a poor proxy for representation priority under a constrained budget. Coverage completeness alone therefore cannot compensate for inefficient allocation, supporting the TRACS principle that selection quality depends on where limited capacity is assigned. Replacing Sobol retrieval with random within-cluster selection also increases both errors under identical quotas and repair. The difference therefore lies in within-region placement. Random retrieval can concentrate samples locally and leave larger continuous-space gaps, whereas structured filling uses the same budget to cover each region more evenly. Thus, appropriate quota allocation is necessary but insufficient: poorly placed samples can still waste capacity through local redundancy. The full configuration also achieves the lowest MASE on more tasks than any ablation variant (Table 16), indicating that this benefit is not confined to a small num-



(a) MASE comparison (b) CRPS comparison

(c) MASE reduction vs. full corpus (d) Ablation study

Figure 4: Panels (a)–(b) report MASE and CRPS per method, panel (c) TRACS's full-corpus MASE reduction, and panel (d) the 50% MOMENT ablation study. BLAST denotes BLAST-style.

ber of datasets. Changing K degrades both aggregate metrics without monotonic improvement as the partition becomes finer. Larger K separates behaviors more precisely but spreads the fixed budget across more clusters; smaller K retains more samples per cluster but merges distinct behaviors. The result therefore reflects a resolution–capacity trade-off: effective selection requires enough partitions to distinguish behaviors without making individual regions too sparsely represented. The No Swap variant isolates a different limitation of Stage 3. It uses the same Stage 3 subset as the full configuration but covers only 93 of the 102 reachable patterns, with all omissions occurring among non-stationary patterns. Stage 3 optimizes continuous filling under the realized quotas, but continuous coverage does not guarantee representation of every discrete seven-bit code. Stage 4 therefore explicitly replaces samples when reachable codes are missing, restoring complete coverage without increasing subset size. This addresses a structural mismatch between continuous-space coverage and discrete reachability, rather than merely adding more data. Together, the ablations identify three complementary failure modes. Quota rebalancing prevents high-frequency regions from dominating the budget, within-cluster retrieval reduces continuous coverage gaps, and final repair eliminates discrete omissions that continuous selection cannot guarantee. Their separate degradations show that these components are not interchangeable: they act at different levels of the selection process and jointly determine whether a limited pretraining budget is efficiently distributed and structurally complete.

6 Conclusion

TRACS treats time-series pretraining data selection as a coverage-constrained allocation problem. Given a fixed selection budget, it redistributes samples across behavioral clusters, selects representative sequences within each cluster, and repairs missing reachable patterns. This procedure preserves the temporal structures targeted by the selection process while keeping the selected distribution close to the rebalanced target. Under the same 10B-token training budget, TRACS subsets outperform both full-corpus pretraining and reduced-data baselines across three architectures. The results indicate that pretraining quality depends on how a limited data budget is distributed and represented, not only on the amount of data retained. We leave two questions for future work: whether the same selection principle transfers to multivariate series and non-forecasting tasks, and how tight the distributional bounds are in practice.

AI USE STATEMENT

Large language models were used to polish manuscript writing and assist with literature search. All research ideas, technical designs, mathematical conclusions, experiments, and the primary manuscript were independently conceived and validated by the authors, who take full responsibility for the work.

ETHICS STATEMENT

This paper uses publicly available benchmark datasets (GIFT-Eval, TIME, and the M4/LOTSA subpool) and does not involve human subjects, personally identifiable information, or the release of new data. We do not foresee any ethical concerns arising from this work.

REPRODUCIBILITY STATEMENT

To support reproducibility, we release an anonymized implementation of TRACS at `https://anonymous.4open.science/r/TRACS-3642`. Algorithmic details are provided in Appendix C, including Algorithm 1 and Algorithms 2–3, while training hyperparameters and configurations are summarized in Appendix E. All experiments use public benchmarks and fixed random seeds. The paired-bootstrap evaluation protocol is described in Section 5.1 and Appendix F.

REFERENCES

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S. Morcos. SemDeDup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*, 2023.

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. GIFT-Eval: A benchmark for general time series forecasting model evaluation. In *NeurIPS 2024 Workshop on Time Series in the Age of Large Models*, 2024.

Theodore W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley, 3rd edition, 2003.

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.

Xiang Ao and Mengru Chen. Efficient data selection for time series forecasting using a lightweight linear proxy framework. *Scientific Reports*, 16(1):16847, 2026. doi: 10.1038/s41598-026-43645-y.

Olivier Bachem, Mario Lucic, and Andreas Krause. Scalable k-means clustering via lightweight coresets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1119–1127. ACM, 2018. doi: 10.1145/3219819.3219973.

Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. Time series FeatuRe extraction on basis of scalable hypothesis tests (tsfresh—A python package). *Neurocomputing*, 307:72–77, 2018. doi: 10.1016/j.neucom.2018.03.067.

Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–73, 1990.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, 2nd edition, 2006. doi: 10.1002/047174882X.

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.

Dan Feldman. Introduction to core-sets: an updated survey. *arXiv preprint arXiv:2011.09384*, 2020.

Ben D. Fulcher and Nick S. Jones. hctsa: A computational framework for automated time-series phenotyping using massive feature extraction. *Cell Systems*, 5(5):527–531, 2017. doi: 10.1016/j.cels.2017.10.001.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16115–16152. PMLR, 2024.

Peter J. Huber and Elvezio M. Ronchetti. *Robust Statistics*. Wiley, 2nd edition, 2009. doi: 10.1002/9780470434697.

Mark E. Johnson, Leslie M. Moore, and Donald Ylvisaker. Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, 26(2):131–148, 1990. doi: 10.1016/0378-3758(90)90122-b.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012. doi: 10.1561/2200000044.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32369–32399. PMLR, 2024.

Carl H. Lubba, Sarab S. Sethi, Philip Knaute, Simon R. Schultz, Ben D. Fulcher, and Nick S. Jones. catch22: CAnonical time-series CHaracteristics: Selected through highly comparative time-series analysis. *Data Mining and Knowledge Discovery*, 33(6):1821–1852, 2019. doi: 10.1007/s10618-019-00647-x.

Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. Training gaussian mixture models at scale via coresets. *Journal of Machine Learning Research*, 18(160):1–25, 2018. URL `https://jmlr.org/papers/v18/15-506.html`.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14(1):265–294, 1978. doi: 10.1007/bf01588971.

Jerzy Neyman. On the two different aspects of the representative method: The method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–625, 1934. doi: 10.2307/2342192.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, 1992. doi: 10.1137/1.9781611970081.

Federico Pennino and Maurizio Gabbrielli. Optimizing the training diet: Data mixture search for robust time series forecasting. In *Proceedings of the 41st ACM/SIGAPP Symposium on Applied Computing*, pp. 1133–1139. ACM, 2026. doi: 10.1145/3748522.3779754.

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. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL `https://arxiv.org/abs/2602.12147`.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Zezhi Shao, Yujie Li, Fei Wang, Chengqing Yu, Yisong Fu, Tangwen Qian, Bin Xu, Boyu Diao, Yongjun Xu, and Xueqi Cheng. BLAST: Balanced sampling time series corpus for universal forecasting models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 2502–2513. ACM, 2025. doi: 10.1145/3711896.3736860.

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 *International Conference on Learning Representations*, 2025.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S. Morcos. Beyond neural scaling laws: Beating power law scaling via data pruning. In *Advances in Neural Information Processing Systems 35*, pp. 19523–19536, 2022. doi: 10.52202/068431-1419.

Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari S. Morcos. D4: Improving LLM pretraining via document de-duplication and diversification. In *Advances in Neural Information Processing Systems 36*, pp. 53983–53995, 2023. doi: 10.52202/075280-2348.

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.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems 36*, pp. 69798–69818, 2023. doi: 10.52202/075280-3059.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. In *International Conference on Learning Representations*, 2025.

A Data Selection Methods

A.1 Method Descriptions

This appendix extends the method comparison and feature rationale in Section 2 by detailing the selection signals of each method and the construction and adaptation of the time-series features.

A.1.1 BLAST

BLAST (Shao et al., 2025) encodes seven types of statistical indicators into 61-dimensional discrete one-hot features computed before pretraining, projects them to two dimensions via UMAP, and divides them into a 100×100 grid. During sampling, it first selects non-empty grids with equal probability, then selects series and windows within a grid, and adopts grid mixup. Its UMAP parameters are adjusted according to the proportion of non-empty grids and the density standard deviation. The adjustment signal comes from the feature distribution of the candidate corpus, and it neither reads the validation error of the prediction model nor updates the selection rule based on downstream performance, so it belongs to the zero-feedback-at-selection-time setting defined in this paper.

A.1.2 Training Diet

Training Diet (Pennino & Gabbrielli, 2026) generates task-agnostic embeddings with MOMENT-1 Large, obtains 36 clusters via K-Means, and searches the cluster weights with Optuna according to the validation MSE, running 100 trials in total, each trial processing a fixed 0.36 billion training tokens, so the selection rule changes with model performance, making it a model-feedback-at-selection-time method.

A.1.3 LP-MIRROR

LP-Mirror (Ao & Chen, 2026) determines the selection priority of samples with the help of a guidance set. The method first trains a channel-independent linear proxy θ_{src} from the source data, then performs one-step adaptation using the validation set or a small number of target-domain samples to obtain $\theta_{\mathrm{adapted}}$, and sorts candidate samples by the change in loss before and after adaptation. Scoring requires two forward computations, whose first-order approximation corresponds to the alignment between the candidate gradient and the average gradient of the guidance set. Because the adaptation direction is determined by the guidance set, the sample scores contain target-related signals, making LP-Mirror guidance-set supervised.

A.1.4 GENERAL DATA SELECTION METHODS

General data-selection research organizes the training corpus from the perspectives of representative-subset construction, embedding deduplication, and data proportion. Coreset (Bachem et al., 2018; Sener & Savarese, 2018; Feldman, 2020) preserves the relevant properties of the full data with a smaller subset. Submodular selection (Nemhauser et al., 1978) optimizes representativeness through marginal gain. D4 (Tirumala et al., 2023) and SemDeDup (Abbas et al., 2023) focus on redundancy in the embedding space. Data pruning (Sorscher et al., 2022) examines the effect of a carefully selected subset on training efficiency. DPP (Kulesza & Taskar, 2012) and minimax/maximin design (Johnson et al., 1990) characterize diversity with determinant and distance criteria, respectively. DoReMi (Xie et al., 2023) and Data Mixing Laws (Ye et al., 2025) study the relationship between data proportion and model performance. These studies provide the methodological background for the coverage objective and quota design in this paper. The comparison between the filling-distance objective and other selection objectives is given in Appendix B.

A.2 FEATURE REPRESENTATION

A.2.1 STL DECOMPOSITION

STL (Cleveland et al., 1990) decomposes a series into trend, seasonal, and residual components based on LOESS, and its robust version reduces the influence of outliers through iterative reweighting. This paper uses STL to extract these components and then constructs statistical features describing the trend, seasonality, and residual behavior to characterize series from different sources and sampling frequencies in LOTSA. The concrete pipeline of decomposition and feature computation is given in Appendix C.

A.2.2 FEATURE RATIONALE

The original definition of TIME (Qiao et al., 2026) includes a signed trend linear coefficient and the average Pearson correlation between seasonal cycles. This paper defines F_2 as the coefficient magnitude, adopts a simplified correlation quantity for F_4, and adjusts part of the computation according to the Stage 0 routing, so they are referred to as TIME-inspired features. The original results of TIME are used to explain the design rationale of the feature family, while the experiments in this paper evaluate the adapted definition and the 27-dimensional representation. catch22, hctsa, and tsfresh (Lubba et al., 2019; Fulcher & Jones, 2017; Christ et al., 2018) also provide time-series statistical representation methods that do not rely on a pre-trained encoder. The complete definitions of the seven-dimensional structured features and the 20-dimensional extended statistical features in this paper are given in Appendix C.

B SELECTION OBJECTIVE

B.1 INTER-CLUSTER QUOTA

This appendix expands the design rationale in Sections 3 and 4. It relates continuous coverage to discrete coverage, explains inter-cluster quota allocation and alternative within-cluster objectives, and compares the problem formulations of TRACS and BLAST.

B.1.1 COVERAGE NOTIONS

This paper characterizes subset coverage at two levels: continuous space and discrete encoding. The six-dimensional filling distance measures the largest coverage hole in the F_1–F_6 bounding box, while the seven-bit reachable-pattern coverage checks whether the selected subset contains the encoding combinations that actually appear in the corpus. Stage 3 improves the former through within-cluster selection, and Stage 4 guarantees the latter through explicit checking and necessary replacement.

These two objectives are related but cannot replace each other. Good filling in continuous space does not directly guarantee that all discrete patterns are represented. In particular, patterns with fewer samples or those near the encoding split threshold may still be missed. Therefore, TRACS takes continuous filling as the selection objective and full coverage of seven-bit reachable patterns as the final output constraint. The coverage metric and the downstream prediction performance are evaluated separately, and a missed pattern does not by itself imply poor downstream performance. The relevant definitions and experimental results are given in Sections 3 and 5 of the main text.

B.1.2 COVERAGE-AWARE REBALANCED QUOTA

Given the coverage objective, the remaining question is how to allocate the fixed subset budget among clusters. TRACS adopts a coverage-aware rebalanced quota so that the inter-cluster allocation serves structural coverage rather than the estimation of overall statistics.

Neyman allocation in stratified sampling aims to reduce the estimation variance of the population mean. Under the conditions of a fixed sample size and the same unit sampling cost in each stratum, the sample quota is allocated in proportion to the product of the stratum size and the within-stratum standard deviation σ_k (Neyman, 1934). Therefore, clusters with larger size and higher within-cluster variation usually receive more quota. This allocation fits its corresponding estimation objective, but it does not directly control the worst-case coverage radius of the selected subset, nor does it guarantee that low-frequency small clusters receive sufficient coverage quota.

TRACS first assigns the base quotas $q_k^{(0)}$ to all clusters. Clusters with insufficient capacity keep all available samples, and their quota deficits are transferred to clusters with spare capacity so that the total remains m (Section 3). The subsequent within-cluster selection uses the capacity-corrected actual quota q_k. This principle limits the occupation of sample quotas by large clusters, and it is not used to reproduce the empirical distribution of the original corpus, nor to judge the importance of each cluster.

B.2 WITHIN-CLUSTER FILLING

B.2.1 SUBMODULAR OBJECTIVES

After the inter-cluster quotas are determined, within-cluster selection must use each limited quota to cover structural variation. TRACS adopts the filling distance, which directly targets the point in the coverage domain farthest from the selected samples. An alternative is to measure overall representativeness through a submodular function.

If a monotone submodular function replaces the filling distance and only the total-sample-size constraint is temporarily retained, the selection problem can be written as

$$\max_{D' \subset D,\, |D'|=m} f(D'), \qquad f \text{ monotone submodular.} \tag{1}$$

For a normalized, non-negative, monotone submodular function, the classical result (Nemhauser et al., 1978) guarantees that the greedy algorithm that incrementally selects the element with the maximum marginal gain achieves an approximation ratio of

$$(1 - 1/e) \approx 0.632 \tag{2}$$

under the cardinality constraint. A typical objective includes the facility-location function.

$$f(D') = \sum_{j \in D} \max_{i \in D'} s_{ij}, \tag{3}$$

where s_{ij} denotes the similarity between samples i and j. This objective characterizes the overall representativeness by accumulating the maximum similarity between each sample and the selected subset. Submodular objectives such as graph cut can also be used to describe specific forms of representativeness or diversity, whose properties depend on the concrete definition.

Facility location focuses on total similarity gain, whereas the filling distance controls the largest coverage hole. For cross-domain time series with an imbalanced pattern distribution, TRACS uses the latter to control worst-case coverage. A separate discrete check determines whether any seven-bit pattern is missing. This choice is specific to the TRACS objective and does not preclude submodular methods for other representativeness or diversity objectives.

B.2.2 Algorithmic Guarantees

The cardinality-constrained problem above does not yet include the inter-cluster quotas of TRACS. Stage 3 first determines the capacity-corrected q_k and then selects samples within each cluster. If a submodular alternative is adopted, the corresponding partition constraints need to be further incorporated. Therefore, the greedy approximation guarantee that only targets the total cardinality constraint cannot be directly applied to the complete pipeline of this paper.

k-center and farthest-point sampling also revolve around the worst-case coverage radius, and they share a similar geometric meaning with the filling objective of this paper. However, the coverage domain, the inter-cluster quotas, and the concrete selection algorithm jointly determine the applicable theoretical guarantee. TRACS uses Sobol-point mapping to complete within-cluster selection, and performs pattern checking and the necessary cross-cluster Swap in Stage 4. The cluster counts after replacement may deviate from the Stage 3 quotas. Therefore, this paper does not directly invoke the constant approximation guarantee of the standard k-center algorithm, but analyzes the relationship among per-cluster filling, sample mapping, and output correction through Section 4 of the main text and Appendix D.

B.2.3 Computational Considerations

Computational cost also affects the choice of the within-cluster implementation. A naive greedy implementation of submodular maximization at $n > 10^7$ requires repeated marginal-gain evaluations over the remaining candidates. TRACS instead generates Sobol targets once and retrieves approximate nearest neighbors through an HNSW index. This avoids repeated global scans and, at the subset sizes used in this paper, reduces the selection time from days to minutes.

B.3 Differences from BLAST

The objectives and constraints above provide the main basis for comparing TRACS and BLAST. Both organize the corpus using statistical features before pretraining, but they differ in partitioning, quota allocation, and final output requirements. Appendix A.1.1 describes the BLAST implementation. This section focuses on the differences relevant to the problem formulation.

B.3.1 Representation

BLAST (Shao et al., 2025) performs hard grid division on the UMAP two-dimensional representation and organizes candidate samples into non-empty grid cells. TRACS, in contrast, applies PCA to the 27-dimensional features, retaining 95% variance, and fits a GMM in the resulting subspace. The GMM posterior probabilities are used for posterior-entropy computation, while the cluster capacity and quotas are still counted by argmax hard labels. Therefore, the two methods differ in both the representation space and the way of organizing the structure. GMM soft probabilities are not equivalent to soft quota allocation.

B.3.2 Quota Allocation

Given its grid partition, BLAST first selects non-empty grids with equal probability, then selects series and windows within a grid. TRACS first determines the actual quota of each cluster, then completes within-cluster selection through Sobol QMC or posterior-entropy guidance, handling inter-cluster quota allocation and within-cluster sample selection separately.

The BLAST-style baseline in the experiments uses the F_1–F_7 seven-dimensional features and removes grid mixup, and it is an adapted implementation of the original method. Therefore, the relevant experiments compare the two selection pipelines implemented in this paper, and the results cannot be attributed solely to the difference between grid division and GMM. The implementation scope is given in Section 5 of the main text and Appendix E.

B.3.3 Final Output Constraint

After quota allocation and within-cluster selection are completed, TRACS further checks the seven-bit reachable patterns in the fixed-size subset and repairs any omission through a Swap while maintaining the total sample size. The equal-probability grid sampling of BLAST does not include the post-hoc check of seven-bit patterns defined in this paper, so the two methods have different output requirements. Under the feasibility conditions of Theorem 2, this check guarantees coverage of all reachable patterns under the given encoding. Therefore, the capacity-corrected rebalanced quota constrains the Stage 3 selection, while the full coverage of seven-bit reachable patterns constrains the final output.

C Implementation Details

This appendix collects the implementation details of TRACS, ordered by pipeline stage. It covers the Stage 0–1 routing and feature definitions, the Stage 2 coreset and weighted EM algorithm, the rationale for the feature-encoding and clustering configuration, and the algorithm pseudocode.

C.1 Stage 0–1 Implementation Details

This subsection corresponds to Sections 4.1–4.2 and gives the Stage 0 routing algorithm and the complete feature definitions of Stage 1. The Stage 0 pipeline is as follows.

The meanings of the routes are as follows. REJECT discards series with an excessively high missing rate. FALLBACK skips STL and sets $T \leftarrow x$, $S \leftarrow 0$, $R \leftarrow 0$, in which case $F_1 = 1$, F_3–$F_5 = 0$, and the decomposition-dependent f_{09}, f_{15}, f_{16} are recorded as NaN. STL performs the complete trend–seasonal–remainder decomposition. STL_NOSSN keeps only the trend and residual, and discards the seasonal term when the seasonal energy ratio is below 1%. Compared with FALLBACK, the residual of STL_NOSSN has already had the trend removed and is more suitable for computing noise-related features. ACF local peaks are used to identify candidate periods. When multiple peaks coexist, the lag with the largest ACF value is taken as the primary period. $\theta = \max(\text{acf_floor}, 2/\sqrt{L})$ accounts for finite-sample noise while enforcing a fixed lower bound. The above rules are consistent with Section 4.1 and determine the computational branch of the features in the following table.

Table 4 lists the 20-dimensional extended features of Stage 1, grouped into five categories. The global-distribution features (f_{01}–f_{05}) summarize the level, spread, and shape of the series. The fluctuation features (f_{06}–f_{08}) characterize the first-difference magnitude. The period-and-autocorrelation features (f_{09}–f_{13}) record the detected period and lag-based autocorrelation. The noise-diagnosis features (f_{14}–f_{17}) assess residual whiteness, signal-to-noise ratio, seasonal energy, and residual spread. The data-quality features (f_{18}–f_{20}) carry the Stage 0 missing-rate and outlier diagnostics together with the logarithmic sequence length.

Table 5 defines the seven-dimensional TIME-inspired structured features used for within-cluster retrieval and seven-bit pattern encoding. F_1–F_2 describe the trend strength and linearity, F_3–F_4 the seasonal strength and correlation, F_5 the residual short-term autocorrelation, F_6 the normalized spectral complexity, and F_7 the binary stationarity indicator of the augmented Dickey–Fuller test.

Algorithm 1 STL Applicability Diagnosis and Adaptive Routing

Require: series x, timestamps t (optional)
Ensure: route $\in$ {REJECT, FALLBACK, STL, STL_NOSSN}, period P, diagnosis record

Hyperparameters: $L_{\min} = 24$, var_floor$= 10^{-10}$, cv_thresh$= 0.05$, acf_floor$= 0.15$, lag_max_factor$= 1/3$
missing rate $mr \leftarrow$ fraction of NaN
if $mr > 0.5$ **then**
  return REJECT // insufficient information, series discarded
end if
$x \leftarrow$ linear interpolation (interior NaN); $x \leftarrow$ nearest valid fill (end NaN)
IQR outlier fraction $\leftarrow \big|\{i : x_i \notin [Q_1 - 1.5\,\mathrm{IQR},\ Q_3 + 1.5\,\mathrm{IQR}]\}\big| / L$
record (mr, IQR outlier fraction) for Stage 1 f_{18} / f_{19}
if $L < L_{\min}$ **then**
  return FALLBACK, $P = 12$ // series too short to reliably estimate period
end if
if $\mathrm{Var}(x) <$ var_floor **then**
  return FALLBACK, $P = 12$ // near-constant series, LOESS smoothing degrades
end if
if t available **then**
  $dt \leftarrow \mathrm{diff}(t)$; $CV_{dt} \leftarrow \sigma(dt)/\mu(dt)$
  if $CV_{dt} >$ cv_thresh **then**
    return FALLBACK, $P = 12$ // irregular interval violates STL assumption
  end if
end if
$dx \leftarrow \mathrm{diff}(x)$ // first-order difference to remove trend
lag_max $\leftarrow \min(\lfloor L/3 \rfloor,\ 500)$
ACF $\leftarrow \mathrm{acf}(dx,\ \mathrm{nlags} = \text{lag_max})$
$\theta \leftarrow \max(\text{acf_floor},\ 2/\sqrt{L})$ // dynamic threshold, Bartlett bound
peaks $\leftarrow \{\mathrm{lag} > 1 : \mathrm{ACF[lag]} > \theta$ and ACF[lag] is a local maximum$\}$
if peaks is not empty **then**
  $P \leftarrow$ lag with the largest ACF value among peaks
  if $L \geq 2P$ **then**
    return STL, P // covers at least 2 full periods, full STL
  else
    return STL_NOSSN, P // period too long, trend-residual only
  end if
else
  return STL_NOSSN, $P = 12$ // no detectable period, trend-residual only
end if

Table 4: 20-dimensional extended features of Stage 1.

ID	Feature	Computation	Category
f_{01}	Mean	$\mathrm{mean}(x)$	Global distribution
f_{02}	Standard deviation	$\mathrm{std}(x)$	Global distribution
f_{03}	Skewness	adjusted Fisher–Pearson skewness	Global distribution
f_{04}	Kurtosis	Fisher kurtosis	Global distribution
f_{05}	Coefficient of variation	$\mathrm{std}/\lvert\mathrm{mean}\rvert$	Global distribution
f_{06}	Mean of first difference	$\mathrm{mean}(\Delta x)$	Fluctuation
f_{07}	Std of first difference	$\mathrm{std}(\Delta x)$	Fluctuation
f_{08}	Difference-to-original ratio	$\mathrm{std}(\Delta x)/\mathrm{std}(x)$	Fluctuation
f_{09}	Detected period	period output by Stage 0	Period and autocorrelation
f_{10}	lag-1 ACF	$\mathrm{ACF}(1)$ of x	Period and autocorrelation
f_{11}	lag-5 ACF	$\mathrm{ACF}(5)$ of x	Period and autocorrelation
f_{12}	lag-12 ACF	$\mathrm{ACF}(12)$ of x	Period and autocorrelation
f_{13}	Mean of top-3 ACF peaks	top-3 ACF peak mean	Period and autocorrelation
f_{14}	Ljung–Box p-value	residual white-noise test	Noise diagnosis
f_{15}	SNR	$\mathrm{Var}(T+S)/\mathrm{Var}(R)$, truncated ≤ 100	Noise diagnosis
f_{16}	Seasonal energy ratio	$\mathrm{Var}(S)/\mathrm{Var}(x)$	Noise diagnosis
f_{17}	Residual std	$\mathrm{std}(R)$	Noise diagnosis
f_{18}	Missing rate	from Stage 0 diagnosis	Data quality
f_{19}	IQR outlier fraction	from Stage 0 diagnosis	Data quality
f_{20}	$\log_{10}$ sequence length	$\log_{10}(L)$	Data quality

The table lists the seven-dimensional definitions used in this paper. The Fisher score and Cohen's d results of TIME (Qiao et al., 2026) serve only to illustrate the original feature family, not to calibrate the adapted formulas.

C.2 Stage 2 Coreset

This subsection details the three stages of the Coreset GMM-EM described in Section 4.3: heuristic score estimation, weighted coreset construction, and weighted EM. Stage A estimates the heuristic

Table 5: Seven-dimensional TIME-inspired structured features.

Feature	Definition used in this paper	Dimension
F_1	$\mathrm{Var}(T)/(\mathrm{Var}(T)+\mathrm{Var}(R))$	Trend strength
F_2	magnitude of the Legendre linear coefficient of T	Trend linearity
F_3	$\mathrm{Var}(S)/(\mathrm{Var}(S)+\mathrm{Var}(R))$	Seasonal strength
F_4	$\mathrm{ACF}_S(1)$	Seasonal correlation
F_5	$\mathrm{ACF}_R(1)$	Residual short-term autocorrelation
F_6	normalized spectral entropy $-\sum_k p_k \log p_k / \log B$	Spectral complexity
F_7	$I(p_{\mathrm{ADF}} < 0.05)$ (binary stationarity indicator)	Stationarity

score with a lightweight GMM. It randomly samples $n_{\mathrm{light}} = 100{,}000$ series to fit a lightweight GMM ($K_{\mathrm{light}} = 2000$), and uses the maximum GMM posterior probability as a heuristic per-sample score $s_i = \max_k p_{\mathrm{light}}(k \mid f_i)$. A value $s_i \approx 1$ indicates a high-confidence assignment (an easy-to-model sample), whereas a low s_i indicates high posterior uncertainty near a cluster boundary. Evaluating this score requires one lightweight-GMM E step over all $n > 10^7$ series. Although this is the most expensive single operation in Stage 2, it is performed once and its output can be stored.

Stage B constructs the weighted coreset by sampling $n_c = 10^5$ points without replacement. The importance weight $w_i \propto 1/(n_c \cdot s_i)$ ensures that the weighted negative log-likelihood on the coreset is an unbiased estimate. Because s_i is a confidence score rather than a formal sensitivity bound, this compression should be regarded as a heuristic acceleration (a rigorous discussion is given in Appendix C.3).

Stage C runs weighted EM. Since scikit-learn's GaussianMixture implementation does not support sample_weight, TRACS implements its own log-space EM with initialization, E and M steps, and convergence checking, avoiding numerical underflow from products of small probabilities.

Input and output. The input is the coreset sample matrix $F_c \in \mathbb{R}^{n_c \times d'}$ and its importance weights $w \in \mathbb{R}^{n_c}$ ($n_c = 10^5$). The output is the trained GMM parameters $\Theta = \{(\pi_k, \mu_k, \Sigma_k)\}_{k=1}^{K}$ (Σ_k is a diagonal matrix), and the posterior-probability matrix $P \in \mathbb{R}^{n \times K}$ of the full data under the converged GMM.

Weighted K-Means++ initialization. The standard K-Means++ selects initial centers by D^2-sampling. In each round the next center is sampled with probability proportional to the squared distance to the nearest already-selected center. The weighted variant modifies the probability distribution to $w_i \cdot D^2(i)$, so that high-weight samples (strongly representative in the coreset) have a greater chance of becoming initial centers, avoiding low-weight noise points dominating the initialization. After sampling K centers, weighted K-Means iterates until convergence (≤ 50 rounds), hard-assigns each sample to the nearest center, and generates the initial responsibilities $r_{ik} = \mathbf{1}[i \in \text{cluster } k]$. To prevent degenerate clusters (a cluster with no sample), r_{ik} is mixed with a uniform distribution, $r_{ik} \leftarrow 0.95 \cdot r_{ik} + 0.05/K$.

Main loop. The weighted EM alternates between the E and M steps for at most 200 rounds and uses the change in weighted log-likelihood as its convergence criterion.

E step (log-space).

for $k = 1..K$ **do**
  $\mathrm{log_prob}[:, k] = \log \pi_k - \frac{1}{2} \sum_j \left[(f_{ij} - \mu_{kj})^2 / \sigma_{kj}^2 + \log(2\pi\, \sigma_{kj}^2) \right]$
end for
$\mathrm{log_prob_max} \leftarrow \max_k$ (row-wise)
$\mathrm{log_sum} \leftarrow \mathrm{log_prob_max} + \log \sum_k \exp(\mathrm{log_prob}[:, k] - \mathrm{log_prob_max})$ // logsumexp
$\log r_{ik} \leftarrow \mathrm{log_prob}[:, k] - \mathrm{log_sum}$
$r_{ik} \leftarrow \exp(\log r_{ik})$

The logsumexp computation first subtracts the maximum, evaluates the exponential sum, and then restores the maximum in log space, avoiding numerical underflow for large K.

M step (weighted sufficient statistics).

$N_k \leftarrow \sum_i w_i \cdot r_{ik}$ // weighted effective sample count
$\mu_k \leftarrow (1/N_k) \sum_i w_i \cdot r_{ik} \cdot f_i$ // weighted mean (d'-dim vector)
$\sigma_k^2 \leftarrow (1/N_k) \sum_i w_i \cdot r_{ik} \cdot (f_i - \mu_k)^2 + \varepsilon$ // weighted var + reg
$\pi_k \leftarrow N_k / \sum_k N_k$

$\varepsilon = 10^{-6}$ (reg_covar) is added to the diagonal covariance to prevent the variance from degenerating to zero.

Convergence criterion. Each round computes the weighted log-likelihood $\ell_t = \sum_i w_i \cdot \log \sum_k \pi_k \cdot \mathcal{N}(f_i \mid \mu_k, \Sigma_k)$. If $|\ell_t - \ell_{t-1}|/|\ell_{t-1}| < \text{tol} = 10^{-3}$ and ℓ_t does not decrease (tolerating 10^{-6} floating-point noise), the iteration terminates early. If the likelihood decreases by more than tol for 3 consecutive rounds, reg_covar is doubled (adaptive regularization), up to 3 times.

Full E step and persistence. After the GMM training on the coreset converges, a single E step (no M step needed) is performed on the full n series with the final parameters Θ to obtain the full posterior $P \in \mathbb{R}^{n \times K}$ and the hard labels $y_i = \text{argmax}_k\, p_{ik}$, persisted as membership.npy (about 4.8GB, float32, and at the design scale $n = 10^7$ this matrix is about 80GB, see Appendix C.3). The GMM parameters Θ are saved as gmm_params.npz (about 30,000 floats), and F_pca is persisted as features_pca.npy (about 1.2GB, float32). The three together constitute the complete output of Stage 2.

Query interface. get_membership(indices) returns, by index, the $\{\text{posterior} \in \mathbb{R}^{|\text{indices}| \times K}$, labels $\in \mathbb{Z}^{|\text{indices}|}$, entropy $\in \mathbb{R}^{|\text{indices}|}\}$ of the subset. The entropy is computed on the fly at query time by $H_i = -\sum_k p_{ik} \log p_{ik} / \log K$ (normalized to $[0, 1]$), matching the narrow-cluster entropy score in Section 4.4. Persisting the full posterior avoids rerunning the E step. Each query takes about 1ms.

C.3 Clustering Configuration

This subsection gives the rationale for four design choices: pattern encoding, the number of clusters, the coreset, and diagonal covariance. In the pattern encoding, F_1–F_6 are binarized by the global median, and F_7 is obtained from the ADF test and participates directly in the encoding. The marginal balance of the median and the 0.5 breakdown point (Cover & Thomas, 2006; Huber & Ronchetti, 2009) support using this encoding as a coarse-grained representative grid relative to the corpus.

Choice of the number of clusters. The diagonal-covariance GMM independently estimates d' variance parameters per cluster. A χ^2 argument implies that a relative error $\leq \delta$ for a single variance parameter requires $n \geq 2/\delta^2$ (Anderson, 2003, Sec. 3.4). Taking $\delta = 0.2$ gives $n/K \geq 50$ as a practical guideline. The Bonferroni correction requires $n/K \gtrsim 100$–150, but it is overly conservative in the PCA space. The variance estimates of the PCA principal components are highly positively correlated, and the union bound overestimates the actual joint estimation error. We use $n/K \geq 50$ as an empirical guideline because (1) the diagonal covariance has only $K \cdot d'$ parameters (not the $K \cdot d'^2$ of full covariance), and (2) EM uses fractional responsibilities rather than estimating each component from hard assignments alone. In the experiments, $K \in \{500, 1000, 2000, 3000\}$ corresponds to $n/K \in [202, 1211]$ (at the experimental scale $n = 605{,}722$), all $\gg 50$, and the degradation of MASE relative to the default $K = 2000$ is only $+0.028$–$+0.041$ (about 3%), and none is statistically significant (Section 5.3). Therefore, the number of clusters has only a moderate effect on selection quality. These results support choosing K between 500 and 2000, with task-specific adjustment.

Coreset scalability (heuristic acceleration). Direct GMM-EM ($K = 2000$) on the full $n > 10^7$ is infeasible. The E step requires $\approx 3 \times 10^{11}$ floating-point operations and the posterior matrix needs 80GB of memory (both estimated at $n = 10^7$). The proper-coreset framework (Bachem et al., 2018; Lucic et al., 2018) (the K-Means lightweight coreset and its GMM extension) compresses the computation to $n_c = 10^5$. The pipeline runs lightweight GMM ($K = 2000$, $n = 100{,}000$) to obtain a heuristic confidence score, then weighted sampling to form a coreset, and finally weighted EM approximating the full negative log-likelihood. This implementation uses the maximum GMM posterior probability $s_i = \max_k p_{\text{light}}(k \mid f_i)$, which is opposite in direction to the σ_i of Bachem (a large s_i indicates high-confidence trivial samples, while a large σ_i indicates hard-to-model boundary samples). Therefore the approximation guarantee $\varepsilon = O(1/\sqrt{n_c})$ cannot be inherited. At $n = 10^7$, the factor $n/n_c \approx 100\times$ should be understood as a compression ratio rather than an approximation guarantee. We therefore treat this compression as an empirically effective heuristic.

Diagonal covariance. Stage 2 uses incremental PCA to decorrelate the features, making the diagonal-covariance assumption easier to satisfy (Section 4.3). The diagonal covariance reduces the parameter scale from $O(K \cdot d'^2)$ of full covariance to $O(K \cdot d')$.

GMM clustering and pattern encoding play different roles. GMM clustering organizes the corpus in the PCA representation and determines the Stage 3 quotas. Pattern encoding checks the Stage 4

Algorithm 2 Sobol QMC within-cluster sampling

Require: cluster member indices *members*, in-cluster $F_1 \ldots F_7$ values $F_{k,7}$, initial quota $q_k^{(0)}$
Ensure: global indices of $q_k^{(0)}$ selected samples

sobol = Sobol($d = 7$, scramble=True)
$q_{\mathrm{pow2}} = 2^{\lceil \log_2 \max\{1, q_k^{(0)}\} \rceil}$
sobol_pts = sobol.random(q_{pow2})[: $q_k^{(0)}$] // ($q_k^{(0)}$, 7), in $[0, 1]^7$
$l = F_{k,7}$.min(axis= 0); $u = F_{k,7}$.max(axis= 0)
sobol_mapped = l+ sobol_pts $\cdot(u - l)$
F_{norm} = L2_normalize($F_{k,7}$)
index = HNSW(F_{norm}, $M = 32$, efConstruction= 200)
selected = {}; result = [ ]
for each s in sobol_mapped **do**
    s_{norm} = L2_normalize(s)
    $k_{nn} = 1$; found = False
    while True **do**
        I = index.search(s_{norm}, k_{nn})
        for idx in I **do**
            if idx not in selected **then**
                selected.add(idx); result.append(idx); found = True; **break**
            end if
        end for
        if found **then**
            break
        end if
        $k_{nn} = \min(k_{nn} \cdot 2,\ n_k)$
    end while
end for
return result[: $q_k^{(0)}$]

output on F_1–F_7. The two divisions need not be aligned, and the cross-cluster Swap may therefore change the final cluster counts. This relationship is consistent with Sections 3–4 of the main text.

C.4 Algorithm Pseudocode

This subsection corresponds to Sections 4.4–4.5. Algorithm 2 implements the Sobol QMC selection for wide clusters, and Algorithm 3 checks and corrects the seven-bit reachable patterns afterwards.

Algorithm 2 fills a wide cluster as follows. It first draws $q_k^{(0)}$ scrambled Sobol points in $[0, 1]^7$ (padding the count to a power of two so that the low-discrepancy grid is well formed), then rescales each point into the cluster's bounding box $[l, u]$ on the seven feature dimensions. Every rescaled point and every cluster member is L_2-normalized, and the members are indexed with an HNSW graph. For each Sobol point in turn, the algorithm looks up its nearest member that has not yet been selected, doubling the neighborhood size whenever the nearest candidates are all already chosen, and appends that member. The first $q_k^{(0)}$ selected indices are returned. The Sobol mapping therefore spreads the selected samples across the seven-dimensional feature space of the cluster, while the nearest-neighbor lookup maps each low-discrepancy probe back onto an actual series.

Algorithm 3 repairs the pattern coverage after Stage 3. It binarizes F_1–F_6 by their global medians (keeping the raw binary F_7), encodes the seven-bit pattern of every series, and identifies the reachable patterns that the current subset still misses. For each missing pattern it draws a new series with that pattern and removes a redundant one. Among the patterns currently appearing at least twice, it chooses the most frequent and, within it, the member with the smallest average cosine distance to its $k = 5$ nearest neighbors, so that the removal loses the least diverse sample. This one-in-one-out loop runs until no reachable pattern is missing and keeps the total size fixed, hence it explicitly verifies coverage. Because replacement can draw from outside the originating cluster, the final GMM cluster counts may deviate from the Stage 3 quotas.

D Proofs of Theorems 1–3

This appendix gives the complete notation and proofs of Theorems 1–3 in the main text. Figure 5 illustrates the six-bit coverage condition of Theorem 1.

Algorithm 3 Stage 4 coverage check and smart Swap

Require: D_{stage3} ($|D_{stage3}| = m$), $F_{1:7}$ (full n series), m
Ensure: D_{final} ($|D_{final}| = m$; complete seven-bit reachable patterns explicitly checked)
medians = median($F_{1:7}$[:, : 6], axis= 0)
patterns = encode_patterns($F_{1:7}$, medians) // F_7 takes its raw binary value
D = set(D_{stage3})
$M_{missing} = P_{reach} \setminus$ set(patterns[list(D)])
if $M_{missing}$ is empty **then**
  return sorted(D)
end if
lookup = build_lookup(patterns)
while $M_{missing}$ is not empty **do**
  p = choose($M_{missing}$); x_{new} = random_choice(lookup[p] $\setminus$ D)
  counts = count_patterns(patterns, D)
  removable = $\{r : \text{counts}[r] \geq 2\}$
  $p_{max} = \text{argmax}_{r \in \text{removable}}(\text{counts}[r])$
  group = $\{i \in D : \text{patterns}[i] = p_{max}\}$
  $x_{remove} = \text{group}[\text{argmin}(\text{kNN_avg_dist}(F_{1:7}[\text{group}],\ k{=}5))]$
  $D = D \cup \{x_{new}\} \setminus \{x_{remove}\}$
  $M_{missing} = P_{reach} \setminus$ set(patterns[list(D)])
end while
return sorted(D)

Notation for the proofs. We write $I = \{1, \ldots, n\}$ for the index set of the full corpus $D = \{x_i\}_{i \in I}$, $C_k \subseteq I$ for the hard GMM clusters ($1 \leq k \leq K$), and $n_k = |C_k|$ for their sizes. Let $\pi_6 : \mathbb{R}^7 \to \mathbb{R}^6$ satisfy $\pi_6(z_1, \ldots, z_7) = (z_1, \ldots, z_6)$, and write $g_i = \pi_6(f_i^{1:7}) = (F_{i1}, \ldots, F_{i6}) \in \mathbb{R}^6$ for the feature point used in all filling and transport arguments. For index sets, set $d(i, j) = \|g_i - g_j\|_2$ and $d(i, S) = \min_{j \in S} d(i, j)$. The corpus diameter is $\Delta = \max_{i,j \in I} d(i, j)$. The per-cluster diameters Δ_k are introduced in Equation (51).

Let

$$\Omega = \prod_{j=1}^{6} [L_j, U_j], \qquad L_j = \min_{i \in I} F_{ij}, \quad U_j = \max_{i \in I} F_{ij}, \tag{4}$$

be the full-corpus bounding box and set $\mathcal{K} := \Omega$. Denote the Stage 3 and final index sets by S_0 and S_f, respectively, and set $S_k = S_0 \cap C_k$ and $q_k = |S_k|$. These q_k are the realized counts after capacity correction and satisfy $\sum_k q_k = m$.

Write δ_a for the Dirac measure at a. For every nonempty finite index set S, let $\mu_S = |S|^{-1} \sum_{i \in S} \delta_{g_i}$. Thus μ_I is the empirical measure of the full corpus, $\mu_k = n_k^{-1} \sum_{i \in C_k} \delta_{g_i}$ is that of cluster k, and $\mu_{S_k} = q_k^{-1} \sum_{j \in S_k} \delta_{g_j}$. The rebalanced target measure preserves each within-cluster law while reweighting clusters by their realized Stage 3 counts.

$$\mu_q = \sum_{k=1}^{K} \frac{q_k}{m} \mu_k. \tag{5}$$

The symbols μ_{S_0} and μ_{S_f} denote the corresponding uniform empirical measures. All Wasserstein distances in this appendix are W_1 under the Euclidean ground cost, and Lipschitz functionals are defined on the compact set $\mathcal{K}$. The encodings c_6, c_7 and pattern sets P_6, P_7 follow Section 3.

Proof of Theorem 1.

Under the theorem assumptions, every C_k is non-empty and $m \geq K$, so every S_k is non-empty. By the finiteness of the corpus, all bounding boxes below are non-empty compact sets.

Let the six-dimensional bounding box of cluster C_k be

$$\Omega_k = \prod_{j=1}^{6} [L_{kj}, U_{kj}], \qquad L_{kj} = \min_{i \in C_k} F_{ij}, \quad U_{kj} = \max_{i \in C_k} F_{ij}. \tag{6}$$

Recall that S_0 is the output after all within-cluster selection, shortage redistribution, and random completion of Stage 3, and define

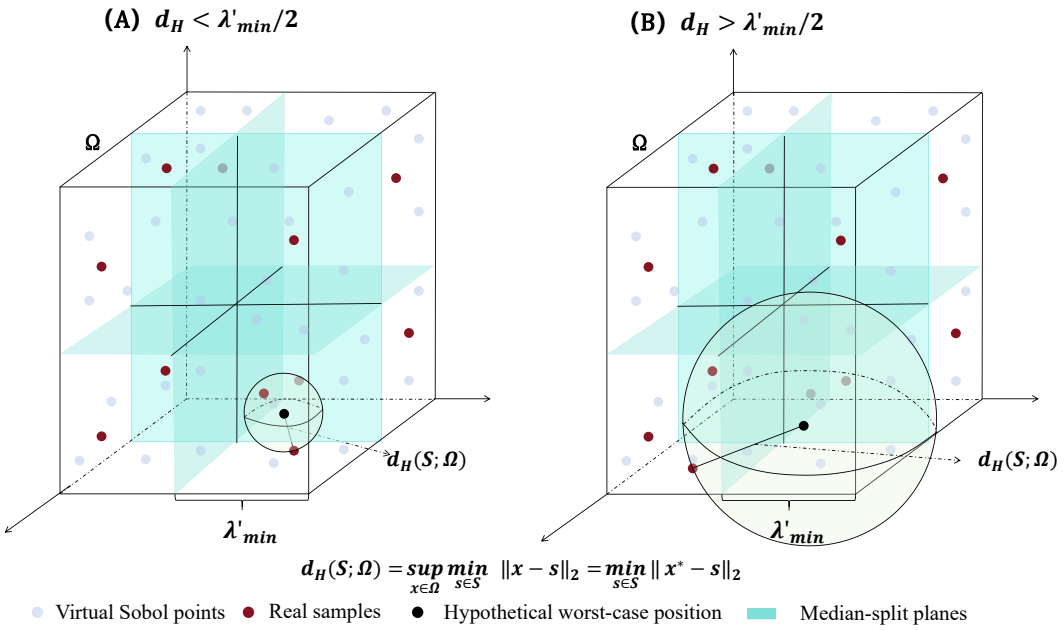


Figure 5: Three-dimensional projection illustration of the six-dimensional continuous coverage condition of Theorem 1. $\lambda'_{\min}(D)$ is the minimum side length of a reachable pattern cell (Equation (19)), and d_H is the one-sided filling distance $h^{\text{fill}}(S_0;\Omega)$.

$$h_k^{\text{fill}} = \sup_{x\in\Omega_k} \min_{i\in S_k} \|x - g_i\|_2. \tag{7}$$

According to the initial within-cluster branch actually executed by Stage 3, decompose the cluster index set disjointly as

$$\{1,\dots,K\} = \mathcal{K}_{\text{Sobol}} \sqcup \mathcal{K}_{\text{ent}} \sqcup \mathcal{K}_{\text{all}}, \tag{8}$$

the three denoting, respectively, the clusters that execute Sobol mapping, posterior-entropy sorting, and direct selection of all available samples.

For $k \in \mathcal{K}_{\text{Sobol}}$, denote by $T_k^{(7)} \subset \mathbb{R}^7$ the seven-dimensional Sobol point set actually generated by the code, and let

$$T_k^{(6)} = \pi_6(T_k^{(7)}) \subset \Omega_k. \tag{9}$$

For $t \in T_k^{(7)}$, denote by $s_k(t) \in C_k$ the index actually returned by the seven-dimensional normalized retrieval. Shortage redistribution and random completion only add samples and do not remove the initially mapped samples. Hence $s_k(t) \in S_k$. Define

$$h_k^{\text{Sobol}} = \sup_{x\in\Omega_k} \min_{u\in T_k^{(6)}} \|x-u\|_2, \qquad r_k = \max_{t\in T_k^{(7)}} \|\pi_6(t) - g_{s_k(t)}\|_2. \tag{10}$$

r_k is the six-dimensional Euclidean error of the actually returned point, and the seven-dimensional retrieval is not required to minimize this error. For clusters that do not execute the Sobol branch, define

$$h_k^{\text{act,fill}} = h_k^{\text{fill}}, \qquad k \in \mathcal{K}_{\text{ent}} \cup \mathcal{K}_{\text{all}}. \tag{11}$$

$h_k^{\mathrm{act,fill}}$ is computed from the final S_k after redistribution, and it is not interpreted as a geometric upper bound implied by the entropy sorting itself. Write uniformly

$$\bar{h}_k = \begin{cases} h_k^{\mathrm{Sobol}} + r_k, & k \in \mathcal{K}_{\mathrm{Sobol}}, \\ h_k^{\mathrm{act,fill}}, & k \in \mathcal{K}_{\mathrm{ent}} \cup \mathcal{K}_{\mathrm{all}}. \end{cases} \tag{12}$$

Define the coverage gap of the cluster bounding boxes relative to the global bounding box

$$\Delta_{\mathrm{gap}} = \sup_{x \in \Omega} \min_{1 \le k \le K} \mathrm{dist}(x, \Omega_k), \qquad \mathrm{dist}(x, \Omega_k) = \inf_{y \in \Omega_k} \|x - y\|_2. \tag{13}$$

For the j-th dimension, the interval splits at the global median $\tilde{F}_j$ into the two half-intervals

$$I_j^0 = [L_j, \tilde{F}_j], \qquad I_j^1 = (\tilde{F}_j, U_j]. \tag{14}$$

For any integer encoding $p \in P_6$, let $b(p) = (b_1(p), \ldots, b_6(p)) \in \{0,1\}^6$ be the unique six-bit vector satisfying

$$p = \sum_{j=1}^{6} 2^{j-1} b_j(p) \tag{15}$$

and define its pattern cell

$$P_p = \prod_{j=1}^{6} I_j^{b_j(p)}. \tag{16}$$

By $b_{ij} = \mathbf{1}\{F_{ij} > \tilde{F}_j\}$ and the open/closed endpoint convention of the intervals above, for every $i \in I$ there is the strict equivalence

$$c_6(i) = p \iff g_i \in P_p. \tag{17}$$

Conventionally, the length of a half-open or closed interval is the difference between its upper and lower endpoints, i.e.,

$$|I_j^0| = \tilde{F}_j - L_j, \qquad |I_j^1| = U_j - \tilde{F}_j. \tag{18}$$

Define the minimum side length of the reachable pattern cells

$$\lambda'_{\min}(D) = \min_{p \in P_6} \min_{1 \le j \le 6} |I_j^{b_j(p)}|. \tag{19}$$

If $\lambda'_{\min}(D) > 0$ and

$$\max_{1 \le k \le K} \bar{h}_k + \Delta_{\mathrm{gap}} < \frac{\lambda'_{\min}(D)}{2}, \tag{20}$$

then the Stage 3 set S_0 covers every six-bit reachable combination in P_6, i.e.,

$$\{c_6(i) : i \in S_0\} = P_6. \tag{21}$$

We first prove the per-cluster filling upper bound. For $k \in \mathcal{K}_{\mathrm{Sobol}}$ and $x \in \Omega_k$, there exists u_x in the finite set $T_k^{(6)}$ such that

$$\|x - u_x\|_2 = \min_{u \in T_k^{(6)}} \|x - u\|_2 \le h_k^{\mathrm{Sobol}}. \tag{22}$$

Take $t_x \in T_k^{(7)}$ such that $\pi_6(t_x) = u_x$. By $s_k(t_x) \in S_k$ and the definition of r_k,

$$\|u_x - g_{s_k(t_x)}\|_2 = \|\pi_6(t_x) - g_{s_k(t_x)}\|_2 \le r_k. \tag{23}$$

Hence, by the triangle inequality,

$$\min_{i \in S_k} \|x - g_i\|_2 \le \|x - g_{s_k(t_x)}\|_2 \le \|x - u_x\|_2 + \|u_x - g_{s_k(t_x)}\|_2 \le h_k^{\mathrm{Sobol}} + r_k. \tag{24}$$

Taking the supremum over x,

$$h_k^{\mathrm{fill}} \le h_k^{\mathrm{Sobol}} + r_k = \bar{h}_k. \tag{25}$$

If $k \in \mathcal{K}_{\mathrm{ent}} \cup \mathcal{K}_{\mathrm{all}}$, then by definition

$$h_k^{\mathrm{fill}} = h_k^{\mathrm{act,fill}} = \bar{h}_k. \tag{26}$$

Therefore, for all k,

$$h_k^{\mathrm{fill}} \le \bar{h}_k. \tag{27}$$

We next prove the global filling upper bound. Write

$$h^{\mathrm{fill}}(S_0; \Omega) = \sup_{x \in \Omega} \min_{i \in S_0} \|x - g_i\|_2. \tag{28}$$

Take any $x \in \Omega$. By the compactness of Ω_k and the finiteness of the number of clusters, there exist k^* and $y^* \in \Omega_{k^*}$ such that

$$\|x - y^*\|_2 = \min_{1 \le k \le K} \mathrm{dist}(x, \Omega_k) \le \Delta_{\mathrm{gap}}. \tag{29}$$

Since S_{k^*} is non-empty and finite, one can take $s^* \in S_{k^*}$ such that

$$\|y^* - g_{s^*}\|_2 = \min_{i \in S_{k^*}} \|y^* - g_i\|_2 \le h_{k^*}^{\mathrm{fill}} \le \max_{1 \le k \le K} \bar{h}_k, \tag{30}$$

and since $S_{k^*} \subset S_0$, the triangle inequality gives

$$\min_{i \in S_0} \|x - g_i\|_2 \le \|x - g_{s^*}\|_2 \le \|x - y^*\|_2 + \|y^* - g_{s^*}\|_2 \le \Delta_{\mathrm{gap}} + \max_{1 \le k \le K} \bar{h}_k. \tag{31}$$

Taking the supremum over x yields

$$h^{\mathrm{fill}}(S_0; \Omega) \le \Delta_{\mathrm{gap}} + \max_{1 \le k \le K} \bar{h}_k. \tag{32}$$

We next prove the separation distance of the pattern cells. Fix $p \in P_6$. By $\lambda'_{\min}(D) > 0$, let

$$x_{p,j}^* = \frac{\inf I_j^{b_j(p)} + \sup I_j^{b_j(p)}}{2}, \qquad x_p^* = (x_{p,1}^*, \ldots, x_{p,6}^*). \tag{33}$$

Then $x_p^* \in P_p$. For any $z \in \Omega \setminus P_p$, at least one coordinate j_0 falls into the opposite half-interval, so

$$\|x_p^* - z\|_2 \geq |x_{p,j_0}^* - z_{j_0}| \geq \frac{|I_{j_0}^{b_{j_0}(p)}|}{2} \geq \frac{\lambda'_{\min}(D)}{2}. \tag{34}$$

We finally prove the pattern coverage (by contradiction). If some $p \in P_6$ is not covered by S_0, then by $c_6(i) = p \Leftrightarrow g_i \in P_p$,

$$g_i \notin P_p, \qquad i \in S_0. \tag{35}$$

By Eq. (34),

$$\min_{i \in S_0} \|x_p^* - g_i\|_2 \geq \frac{\lambda'_{\min}(D)}{2}. \tag{36}$$

Hence

$$h^{\text{fill}}(S_0; \Omega) \geq \frac{\lambda'_{\min}(D)}{2}. \tag{37}$$

On the other hand, Eqs. (20) and (32) give

$$h^{\text{fill}}(S_0; \Omega) \leq \Delta_{\text{gap}} + \max_{1 \leq k \leq K} \bar{h}_k < \frac{\lambda'_{\min}(D)}{2}, \tag{38}$$

contradicting Eq. (37), so

$$P_6 \subseteq \{c_6(i) : i \in S_0\}. \tag{39}$$

Since $S_0 \subseteq I$,

$$\{c_6(i) : i \in S_0\} \subseteq P_6. \tag{40}$$

Combining the two yields (21). This completes the proof.

Figure 5 is a low-dimensional illustration, and the Sobol points correspond only to $\mathcal{K}_{\text{Sobol}}$. This theorem handles only the six-bit patterns, and the seven-bit coverage is guaranteed by Theorem 2.

Proof of Theorem 2.

Let the number of seven-bit reachable patterns be $R = |P_7|$, and let the initial number of patterns missing from the Stage 3 output be

$$s_0 = |P_7 \setminus \{c_7(i) : i \in S_0\}|. \tag{41}$$

For an index set $S \subseteq I$, recall $d(i, S) = \min_{j \in S} \|g_i - g_j\|_2$ and let $h_D(S) = \max_{i \in I} d(i, S)$ be the discrete covering radius over the finite corpus D. It is the finite-sample counterpart of the continuous bounding-box filling distance $h^{\text{fill}}(S; \Omega)$ of Theorem 1, satisfying $h_D(S) \leq h^{\text{fill}}(S; \Omega)$.

Denote $c_k = |S_f \cap C_k|$. The t-th Swap removes r_t and adds a_t, and define $\rho_t = d(r_t, S_{t-1} \setminus \{r_t\})$.

We first prove feasibility. If a set of size m covers R patterns, then by the pigeonhole principle $m \geq R$. Conversely, taking one representative from each $p \in P_7$ and then completing to m samples by $m \leq n$ gives a feasible set. Therefore, under $m \leq n$, there exists a subset of size m covering all of P_7 if and only if $m \geq R$.

We next prove the pattern repair and finite termination. Let the current set S cover $r < R$ patterns, the covered set be $Q(S)$, and the pattern counts be $N_p(S)$. Since $|S| = m$,

$$\sum_{p \in Q(S)} N_p(S) = m. \tag{42}$$

If all counts were 1, then $m = r < R$, contradicting $m \geq R$, so there exists a safely removable pattern with count at least 2. Take $p_{\text{miss}} \in P_7 \setminus Q(S)$. By reachability, there exists $a \notin S$ such that $c_7(a) = p_{\text{miss}}$. Take r_0 from a safe pattern and let $S' = (S \setminus \{r_0\}) \cup \{a\}$. The deletion creates no new miss, and the addition adds one new pattern, so $Q(S') = Q(S) \cup \{p_{\text{miss}}\}$, and the number of misses decreases by exactly 1. Therefore the algorithm terminates after exactly $s = s_0$ Swaps, and

$$|S_f| = m, \qquad \{c_7(i) : i \in S_f\} = P_7. \tag{43}$$

We next prove that the quota drift is bounded. Let A_k and B_k denote the numbers of samples added to and removed from cluster k, respectively. Then

$$c_k = q_k + A_k - B_k, \qquad \sum_k A_k = \sum_k B_k = s. \tag{44}$$

Hence $\sum_k c_k = m$, $|A_k - B_k| \leq s$, and $\sum_k |c_k - q_k| \leq \sum_k (A_k + B_k) = 2s$, i.e.,

$$\sum_k c_k = m, \qquad |c_k - q_k| \leq s, \qquad \sum_k |c_k - q_k| \leq 2s. \tag{45}$$

We further prove that the coverage-radius increment is bounded. For step t, removing r_t from S_{t-1}, write $S^- = S_{t-1} \setminus \{r_t\}$, so that $\rho_t = d(r_t, S^-)$ by definition. The distance can increase only when the nearest representative is r_t. Take $r' \in S^-$ such that $d(r_t, r') = \rho_t$, then

$$d(i, S^-) \leq d(i, r') \leq d(i, r_t) + d(r_t, r') \leq h_D(S_{t-1}) + \rho_t. \tag{46}$$

Adding a sample does not increase the distance, so $h_D(S_t) \leq h_D(S_{t-1}) + \rho_t$, and iterating gives

$$h_D(S_f) \leq h_D(S_0) + \sum_{t=1}^{s} \rho_t. \tag{47}$$

We finally prove that the movement of the sample distribution is bounded. Adjacent sets share $m-1$ samples. Fixing the common mass and only transporting the mass $1/m$ of the deleted point to the added point,

$$W_1(\mu_{S_{t-1}}, \mu_{S_t}) \leq \frac{1}{m} \|g_{r_t} - g_{a_t}\|_2. \tag{48}$$

Applying the triangle inequality to each step and bounding the distance by Δ,

$$W_1(\mu_{S_0}, \mu_{S_f}) \leq \frac{1}{m} \sum_{t=1}^{s} \|g_{r_t} - g_{a_t}\|_2 \leq \frac{s\Delta}{m}. \tag{49}$$

The cosine redundancy score only determines the sample to remove within the safe pattern and does not affect the proof.

This completes the proof.

Proof of Theorem 3.

We first fix the notation. For each cluster, choose a nearest representative $A_k(i) \in \operatorname{argmin}_{j \in S_k} d(i, j)$, and define

$$w_{kj} = \frac{1}{n_k}|\{i \in C_k : A_k(i) = j\}|, \qquad \beta_k = \frac{1}{2}\sum_{j \in S_k}\left|w_{kj} - \frac{1}{q_k}\right|, \tag{50}$$

$$R_k = \frac{1}{n_k}\sum_{i \in C_k} d(i, S_k), \qquad \Delta_k = \max_{i,j \in C_k} d(i,j), \tag{51}$$

$$\varepsilon_0 = \sum_{k=1}^{K} \frac{q_k}{m}(R_k + \Delta_k \beta_k), \qquad \beta_{\text{quota}} = \frac{1}{2}\sum_{k=1}^{K}\left|\frac{n_k}{n} - \frac{q_k}{m}\right|. \tag{52}$$

Under the conditions of Theorem 2, let

$$\varepsilon_f = \varepsilon_0 + \frac{1}{m}\sum_{t=1}^{s} \|g_{r_t} - g_{a_t}\|_2. \tag{53}$$

We next prove the per-cluster Wasserstein upper bound. For cluster k define $\nu_k = \sum_{j \in S_k} w_{kj}\delta_{g_j}$. Transporting the mass $1/n_k$ of each g_i to $g_{A_k(i)}$ costs R_k, so $W_1(\mu_k, \nu_k) \le R_k$. The total mass needed to adjust from ν_k to μ_{S_k} is $\frac{1}{2}\sum_{j \in S_k}|w_{kj} - 1/q_k| = \beta_k$, and the within-cluster distance does not exceed Δ_k, so $W_1(\nu_k, \mu_{S_k}) \le \Delta_k \beta_k$. By the triangle inequality,

$$W_1(\mu_k, \mu_{S_k}) \le R_k + \Delta_k \beta_k. \tag{54}$$

We next prove the global Wasserstein upper bound. For each k take the above coupling π_k, and let $\pi = \sum_{k=1}^{K} \frac{q_k}{m}\pi_k$. Its marginals are μ_q and μ_{S_0}, and the total cost does not exceed ε_0, so $W_1(\mu_q, \mu_{S_0}) \le \varepsilon_0$. By Theorem 2,

$$W_1(\mu_{S_0}, \mu_{S_f}) \le \frac{1}{m}\sum_{t=1}^{s} \|g_{r_t} - g_{a_t}\|_2. \tag{55}$$

Combining with the above through the triangle inequality gives

$$W_1(\mu_q, \mu_{S_f}) \le \varepsilon_f \le \varepsilon_0 + \frac{s\Delta}{m}. \tag{56}$$

We further prove the bound relative to the full corpus. We have $\mu_I = \sum_k \frac{n_k}{n}\mu_k$ and $\mu_q = \sum_k \frac{q_k}{m}\mu_k$. The within-cluster distributions of the two measures are the same, and the transported mass corresponding to the weight difference is β_{quota}, so $W_1(\mu_I, \mu_q) \le \Delta\beta_{\text{quota}}$, and combining with (56) gives

$$W_1(\mu_I, \mu_{S_f}) \le \Delta\beta_{\text{quota}} + \varepsilon_f. \tag{57}$$

We finally prove the Lipschitz functional stability. After fixing $x_0 \in \mathcal{K}$, $\text{Lip}_0(\mathcal{K})$ is a Banach space under the Lipschitz norm. If (φ_n) is a Cauchy sequence, then by $\varphi_n(x_0) = 0$,

$$\|\varphi_n - \varphi_m\|_\infty \le \text{diam}(\mathcal{K})\|\varphi_n - \varphi_m\|_{\text{Lip}}. \tag{58}$$

Hence it converges uniformly to a function φ satisfying $\varphi(x_0) = 0$. For any $\epsilon > 0$, take N such that $\|\varphi_n - \varphi_m\|_{\text{Lip}} < \epsilon$ when $n, m \ge N$. Fixing $n \ge N$ and letting $m \to \infty$ gives

$$\frac{|(\varphi_n - \varphi)(x) - (\varphi_n - \varphi)(y)|}{\|x - y\|_2} \le \epsilon. \tag{59}$$

Table 6: Hyperparameters and configuration reference.

Hyperparameter	Value	Note
Context window	512	TIME benchmark standard
Prediction length	96	same as above
Sliding stride	128	window overlap
Batch size	512	unified across the three models, V100 32GB
Eval batch size	64	GIFT-Eval
Learning rate	1×10^{-4}	linear warmup 5% $\rightarrow$ cosine decay
Warmup ratio	5%	linear warmup
Weight decay	0.01	AdamW
Max grad norm	1.0	gradient clipping
Training tokens/run	10B	fixed token budget
Mixed precision	AMP	bfloat16/float16 automatic
TRACS K	2000	default number of clusters
TRACS ACF floor	0.15	Stage 0 period-detection threshold
TRACS PCA variance retained	0.95	incremental PCA
TRACS STL L_max	10,000	maximum length of STL decomposition
TRACS quota_mode	equal	rebalanced quota
TRACS use_sobol	True	Sobol QMC
Bootstrap resamples	10,000	paired bootstrap
Bootstrap CI	95%	
GIFT-Eval tasks	28	
Eval max series per task	50	GIFT-Eval

Taking the supremum over $x \neq y$ gives $\|\varphi_n - \varphi\|_{\mathrm{Lip}} \to 0$, so $\mathrm{Lip}_0(\mathcal{K})$ is complete. For any coupling $\pi \in \Pi(\mu, \nu)$,

$$\left| \int \varphi \, d\mu - \int \varphi \, d\nu \right| = \left| \int (\varphi(x) - \varphi(y)) \, d\pi \right| \leq \|\varphi\|_{\mathrm{Lip}} \int \|x - y\|_2 \, d\pi. \tag{60}$$

Taking the infimum over couplings and substituting (56) gives

$$\left| \int \varphi \, d\mu_q - \int \varphi \, d\mu_{S_f} \right| \leq \|\varphi\|_{\mathrm{Lip}} \varepsilon_f. \tag{61}$$

This completes the proof.

E Experimental Hyperparameters

Table 6 lists the shared training and evaluation hyperparameters (context window, prediction length, batch size, learning-rate schedule, and the fixed 10B-token budget), the TRACS-specific selection parameters (the number of clusters K, the Stage 0 ACF floor, the retained PCA variance, the rebalanced quota mode, and the Sobol switch), and the statistical conventions (10,000 bootstrap resamples and the 95% interval).

This appendix corresponds to Section 5 and summarizes the baseline implementations, the statistical conventions, the computational scale, and the coverage definitions. The original BLAST (Shao et al., 2025) constructs a 100×100 grid in a two-dimensional UMAP embedding and adopts grid mixup. The BLAST-style baseline in this paper uses the F_1–F_7 seven-dimensional features and removes grid mixup. TRACS uses the 27-dimensional PCA/GMM representation and seven-dimensional within-cluster retrieval analyzed through a six-dimensional projection, so the relevant results compare two complete implementations.

Statistical testing procedure. Among the 28 GIFT-Eval candidate tasks, the MASE of 7 tasks is undefined, and covid_deaths is excluded. Table 2 computes the geometric mean over the remaining 20 stable tasks. The per-task arithmetic differences, Win/Tie/Loss, and paired bootstrap are likewise based on these 20 stable tasks. The bootstrap first averages the five seeds for each task and then resamples with replacement 10,000 times by task. The 95% interval describes the fluctuation of the task composition and does not include seed-level uncertainty, and neither the p-values nor the intervals are corrected.

Table 7: Uncorrected p-values of per-task MASE differences on the 20 numerically stable tasks (paired bootstrap, 10,000 resamples, descriptive reference only).

	TRACS-25 vs.			TRACS-50 vs.		
Opponent	MOMENT	PatchTST	Timer	MOMENT	PatchTST	Timer
BLAST-style	0.554	0.115	0.004	0.055	0.574	0.048
Random	0.199	0.017	0.506	0.788	0.025	0.231
Stratified	0.285	0.064	0.028	0.016	0.015	0.218
Full Corpus	0.525	0.054	0.021	0.456	0.047	0.055

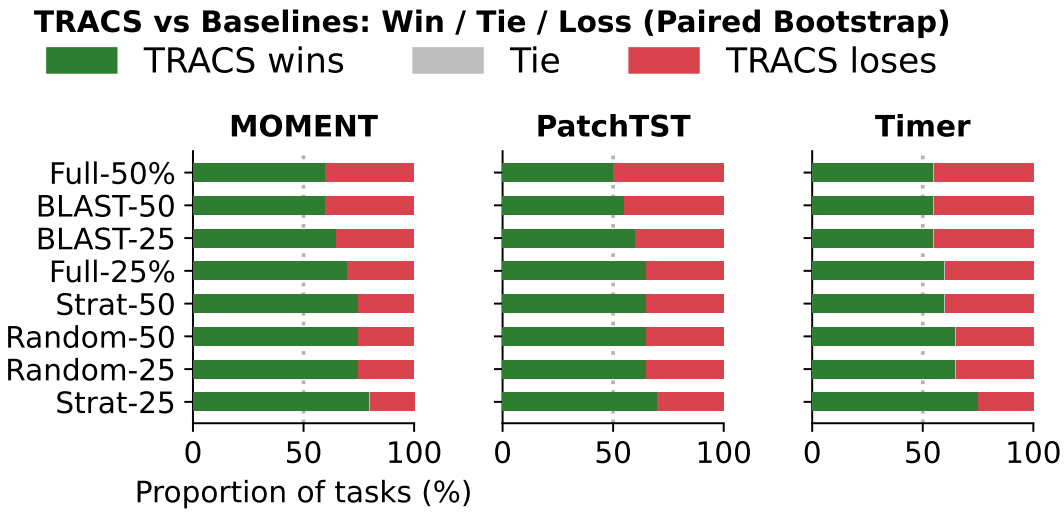


Figure 6: Win, tie, and loss counts of TRACS against each baseline. BLAST denotes BLAST-style.

Experimental scale. The main experiments have 135 jobs and the ablation experiments have 30 jobs, 165 jobs in total. On a V100 32GB, a single MOMENT, PatchTST, and Timer job takes about 1.6 h, 1.5 h, and 0.7 h respectively, totaling about 223 GPU-hours, or about 245 h after adding a 10% buffer. With 2×V100 it takes about 5 days.

Under the 128-denominator convention for pattern coverage, F_1–F_6 are binarized by the global median and F_7 takes its raw binary value, producing $2^7 = 128$ theoretical pattern encodings, of which 26 combinations do not exist in the LOTSA subpool, so there are 102 reachable patterns. When taking 128 as the denominator, the coverage of TRACS is equivalent to $102/128 \approx 79.7\%$ (the abs (covered/128) column of Table 14 in Appendix F), which is used only as an auxiliary convention when discussing the theoretical pattern space.

F Paired Bootstrap Details

This appendix corresponds to Sections 5.2–5.3 and reports the paired bootstrap and per-task MASE over the 20 stable tasks (covid_deaths excluded). Table 7 collects the uncorrected per-task p-values across the three architectures and both budgets as a compact, descriptive summary. In the paired-bootstrap tables below, the Mean diff column reports the per-task arithmetic difference (negative favors TRACS, whose MASE is lower), the 95% interval and p-value come from the 10,000-resample paired bootstrap, and the Win/Tie/Loss column counts the tasks on which TRACS wins, ties, or loses to the opponent. The per-task MASE tables list every method with the best value per task in bold. In the per-task MASE tables, column headers abbreviate the baselines as Rand (Random), Strat (Stratified), and BLAST (BLAST-style), with the -25/-50 suffix denoting the subset percentage. Figure 6 visualizes the per-baseline Win/Tie/Loss counts of TRACS, and Figure 7 scatters, for each task, the MASE of TRACS against that of each baseline, so that points below the diagonal mark tasks on which TRACS attains the lower MASE.

Table 8: Paired bootstrap for MOMENT (TRACS vs each baseline, MASE).

Comparison	Mean diff	95% CI	p-value	Win/Tie/Loss
TRACS-25 vs BLAST-style-25	-0.0164	$[-0.0690, +0.0408]$	0.5542	13/0/7
TRACS-25 vs Full	-0.0366	$[-0.1440, +0.0903]$	0.5252	14/0/6
TRACS-25 vs Random-25	-0.0591	$[-0.1558, +0.0316]$	0.1994	15/0/5
TRACS-25 vs Stratified-25	-0.0256	$[-0.0673, +0.0241]$	0.2854	16/0/4
TRACS-50 vs BLAST-style-50	-0.0565	$[-0.1514, +0.0005]$	0.0550	12/0/8
TRACS-50 vs Full	-0.0489	$[-0.1808, +0.0602]$	0.4558	12/0/8
TRACS-50 vs Random-50	-0.0171	$[-0.1365, +0.1055]$	0.7882	15/0/5
TRACS-50 vs Stratified-50	-0.0617	$[-0.1379, -0.0072]$	0.0156	15/0/5

Table 9: Per-task MASE details for MOMENT (mean over 5 seeds).

Task	Rand-25	Rand-50	Strat-25	Strat-50	BLAST-25	BLAST-50	TRACS-25	TRACS-50	Full
electricity_15t	0.6020	0.5995	0.5985	0.5946	0.5978	0.5972	**0.5936**	0.5993	0.6055
electricity_h	0.8256	0.8411	0.8586	0.8606	0.8226	0.8499	**0.7535**	0.8337	0.8023
hierarchical_sales_w	**0.8763**	0.8821	0.8950	0.8911	0.9000	0.8973	0.8906	0.8764	0.8830
hospital	1.0425	**1.0054**	1.0272	1.0263	1.0439	1.0776	1.0415	1.0236	1.0203
loop_seattle_5t	0.6578	0.6696	0.6571	0.6594	0.6612	0.6605	0.6558	**0.6502**	0.6540
loop_seattle_d	0.9502	0.9851	0.9620	0.9322	0.9957	0.9863	**0.9055**	0.9440	0.9785
loop_seattle_h	0.7475	0.8071	0.7414	0.7340	0.7467	0.7247	**0.6970**	0.7278	0.7027
m4_daily	2.2313	2.1137	2.3707	2.3347	2.2791	**2.1124**	2.1270	2.2028	2.1176
m4_hourly	0.3107	0.3269	0.3295	0.3367	0.3344	0.3186	**0.2900**	0.2923	0.3360
m4_monthly	1.4282	1.2748	1.4429	1.4022	1.3090	**1.2265**	1.3165	1.2516	1.3602
m4_quarterly	2.6504	2.2424	2.0444	2.1140	2.3964	2.3398	2.0281	**1.4638**	2.5021
m4_weekly	**1.2213**	1.3131	1.3123	1.2971	1.2814	1.3289	1.2684	1.2408	1.3027
m4_yearly	2.4335	2.2681	2.5920	2.5474	2.8495	2.5688	2.9369	2.5786	**2.0568**
m_dense_d	0.8291	0.7846	0.7955	0.8027	0.7688	**0.7646**	0.7743	0.7745	0.7887
m_dense_h	0.6262	0.6416	**0.5921**	0.6442	0.6197	0.6332	0.5962	0.6145	0.6018
solar_10t	4.4320	3.7274	3.9511	4.3767	**3.5755**	4.4806	3.9267	4.4998	4.3678
solar_d	**0.9393**	0.9529	0.9793	0.9789	0.9689	0.9748	0.9723	0.9669	0.9774
solar_h	3.5549	3.8367	3.5438	3.6356	**3.3510**	3.5214	3.4097	3.3980	3.8504
sz_taxi_15t	0.7785	0.7789	0.7792	**0.7762**	0.7773	0.7771	0.7798	0.7778	0.7766
sz_taxi_h	0.6484	0.6494	0.6437	0.6481	0.6544	0.6480	**0.6411**	0.6417	0.6518

F.1 MOMENT

Table 8 reports the paired bootstrap and Table 9 the per-task MASE for the MOMENT backbone.

F.2 PATCHTST

Table 10 reports the paired bootstrap and Table 11 the per-task MASE for the PatchTST backbone.

F.3 TIMER

Table 12 reports the paired bootstrap and Table 13 the per-task MASE for the Timer backbone.

F.4 ABLATION (MOMENT, 50% SUBSET)

The ablation isolates each design component of TRACS on the MOMENT backbone with a 50% subset, under the same protocol as the main experiments (5 seeds, fixed 10B-token budget, AdamW, Section 5.1). The full configuration is the reference. It uses GMM clustering with $K = 2000$, Sobol in-cluster filling, coverage-aware rebalanced quota, and the Stage 4 Swap. Each variant changes one component and keeps the other three fixed. The $K \in \{500, 1000, 3000\}$ variants probe the sensitivity of GMM clustering, and their rationale is in Appendix C.3. The In-cluster random variant replaces the Sobol mapping with uniform random sampling, ablating the seven-dimensional Sobol mapping-and-retrieval rule analyzed through the six-dimensional projection. The Proportional quota variant removes the coverage-aware rebalancing so that each cluster selects in proportion to its size, ablating the rebalancing quota. The No Swap variant disables the Stage 4 pattern-repair loop and keeps the Stage 3 subset directly, ablating the seven-bit pattern checking.

Table 10: Paired bootstrap for PatchTST (TRACS vs each baseline, MASE).

Comparison	Mean diff	95% CI	p-value	Win/Tie/Loss
TRACS-25 vs BLAST-style-25	-0.1175	$[-0.2901, +0.0214]$	0.1148	10/0/10
TRACS-25 vs Full	-0.2387	$[-0.6453, +0.0013]$	0.0538	13/0/7
TRACS-25 vs Random-25	-0.2727	$[-0.6399, -0.0193]$	0.0170	14/0/6
TRACS-25 vs Stratified-25	-0.1392	$[-0.3291, +0.0053]$	0.0644	13/0/7
TRACS-50 vs BLAST-style-50	-0.0251	$[-0.1147, +0.0626]$	0.5740	12/0/8
TRACS-50 vs Full	-0.2049	$[-0.5177, -0.0018]$	0.0468	11/0/9
TRACS-50 vs Random-50	-0.0908	$[-0.1952, -0.0079]$	0.0248	13/0/7
TRACS-50 vs Stratified-50	-0.2559	$[-0.5975, -0.0210]$	0.0148	13/0/7

Table 11: Per-task MASE details for PatchTST (mean over 5 seeds).

Task	Rand-25	Rand-50	Strat-25	Strat-50	BLAST-25	BLAST-50	TRACS-25	TRACS-50	Full
electricity_15t	1.0393	**1.0187**	1.0453	1.0438	1.0366	1.0832	1.1118	1.0476	1.0240
electricity_h	0.8091	0.8089	0.8377	**0.7805**	0.7907	0.7995	0.7895	0.8039	0.7836
hierarchical_sales_w	0.8976	0.8938	0.8786	0.9435	0.9570	0.9093	0.8847	0.8997	**0.8769**
hospital	3.8485	3.5881	**3.4117**	3.9069	3.5829	3.5387	3.4398	3.5231	3.9830
loop_seattle_5t	2.4481	2.4842	**2.4344**	2.5197	2.4884	2.4804	2.5398	2.4973	2.4625
loop_seattle_d	1.5680	1.6664	1.7945	1.6450	2.4636	1.5920	**1.2129**	1.3810	1.2681
loop_seattle_h	0.7959	**0.7630**	0.7696	0.8303	0.8573	0.7760	0.7632	0.8594	0.7710
m4_daily	7.9237	7.8639	7.9848	**7.4985**	7.7413	8.3356	8.1075	7.6986	8.1665
m4_hourly	0.2886	0.2856	0.3137	0.3002	0.2849	0.3131	0.2874	**0.2784**	0.2920
m4_monthly	2.3219	2.3684	2.2223	2.4286	2.2401	2.3508	2.4205	**2.2001**	2.3342
m4_quarterly	13.7124	12.5408	12.1785	14.5141	11.6286	11.2150	**10.7032**	11.7079	14.4389
m4_weekly	1.7810	**1.7089**	1.8124	1.9696	1.7277	1.8097	1.7655	1.8366	1.7760
m4_yearly	8.8424	7.4991	8.1006	8.5895	7.6827	7.2137	7.2590	**7.0971**	7.7880
m_dense_d	**0.7047**	0.7241	0.7324	0.7209	0.7193	0.7171	0.7208	0.7196	0.7264
m_dense_h	0.5866	0.6104	0.6052	0.6081	0.5923	0.5939	0.5999	**0.5841**	0.6117
solar_10t	2.8743	2.6210	2.8846	2.7259	2.9321	**2.5013**	2.6631	2.7617	2.5361
solar_d	1.0099	1.0025	0.9890	0.9750	0.9762	0.9590	0.9996	**0.9273**	0.9640
solar_h	4.3076	4.3514	4.0866	4.1015	**3.9569**	4.3005	4.0502	4.1669	4.2853
sz_taxi_15t	0.8018	0.8040	0.8109	0.8026	0.8035	0.8029	**0.8000**	0.8045	0.8026
sz_taxi_h	0.6460	0.6421	0.6437	0.6419	0.6406	0.6398	0.6346	**0.6343**	0.6363

Table 14 compares the seven-bit reachable-pattern coverage of the methods and the 50% ablation variants. Table 15 reports the paired bootstrap of each ablation variant against the TRACS-50 control, and Table 16 lists their per-task MASE, with the best value per task in bold.

Table 12: Paired bootstrap for Timer (TRACS vs each baseline, MASE).

Comparison	Mean diff	95% CI	p-value	Win/Tie/Loss
TRACS-25 vs BLAST-style-25	-0.1724	$[-0.4302, -0.0199]$	0.0040	13/0/7
TRACS-25 vs Full	-0.0858	$[-0.1857, -0.0098]$	0.0208	12/0/8
TRACS-25 vs Random-25	$+0.0508$	$[-0.0502, +0.1992]$	0.5062	11/0/9
TRACS-25 vs Stratified-25	-0.1943	$[-0.4804, -0.0091]$	0.0278	12/0/8
TRACS-50 vs BLAST-style-50	-0.1016	$[-0.2606, -0.0005]$	0.0482	11/0/9
TRACS-50 vs Full	-0.1677	$[-0.4306, +0.0018]$	0.0546	15/0/5
TRACS-50 vs Random-50	-0.0498	$[-0.1399, +0.0290]$	0.2306	11/0/9
TRACS-50 vs Stratified-50	-0.1407	$[-0.4185, +0.0427]$	0.2184	13/0/7

Table 13: Per-task MASE details for Timer (mean over 5 seeds).

Task	Rand-25	Rand-50	Strat-25	Strat-50	BLAST-25	BLAST-50	TRACS-25	TRACS-50	Full
electricity_15t	0.6684	0.6581	**0.6539**	0.6600	0.6544	0.6598	0.6718	0.6603	0.6603
electricity_h	0.9501	0.9400	0.9337	0.9393	0.9374	0.9469	**0.9225**	0.9274	0.9517
hierarchical_sales_w	0.8433	0.8729	0.8788	0.8714	0.8992	0.8694	**0.8389**	0.8489	0.8835
hospital	2.1052	2.0251	2.3423	2.0735	2.2689	1.9729	**1.9573**	2.0102	2.0376
loop_seattle_5t	2.4143	2.5126	**2.3035**	2.4046	2.5479	2.4298	2.5162	2.4511	2.3890
loop_seattle_d	1.1537	**1.0019**	1.2017	1.0491	1.0925	1.2029	1.0066	1.0153	1.1264
loop_seattle_h	0.8610	0.9202	0.8673	0.9094	**0.8120**	0.8891	0.9180	0.8911	0.8876
m4_daily	6.0155	5.8761	6.0568	**5.6695**	6.0109	6.0611	6.0133	6.2432	5.9265
m4_hourly	0.3443	0.3570	0.3750	0.3659	0.3522	0.3406	**0.3217**	0.3417	0.3590
m4_monthly	1.7401	1.6619	1.7664	**1.5634**	1.8060	1.7195	1.6749	1.7378	1.8369
m4_quarterly	5.9991	6.3363	9.7029	8.1568	9.6367	7.1674	7.2413	**5.7549**	8.0515
m4_weekly	1.7079	1.6382	**1.6370**	1.6522	1.6780	1.6433	1.7303	1.6461	1.6942
m4_yearly	6.0929	6.4711	7.1449	6.5464	6.6469	6.2050	6.4037	**5.9997**	6.7278
m_dense_d	0.7485	0.7508	0.7547	0.7624	0.7549	0.7626	0.7420	**0.7188**	0.7650
m_dense_h	0.5612	0.5751	0.5545	0.5536	0.5676	0.5619	0.6337	0.5768	**0.5494**
solar_10t	3.0086	3.2147	3.2433	3.2039	3.0712	3.1968	**2.8720**	2.9174	3.1615
solar_d	0.9279	0.9273	**0.9132**	0.9362	0.9215	0.9403	0.9525	0.9242	0.9318
solar_h	3.6010	3.3769	**3.3189**	3.6223	3.5481	3.5786	3.3420	3.4548	3.5348
sz_taxi_15t	0.7849	0.7836	**0.7816**	0.7821	0.7833	0.7860	0.7863	0.7847	0.7827
sz_taxi_h	0.6442	0.6438	0.6438	**0.6403**	0.6451	0.6457	0.6422	0.6440	0.6452

Table 14: Coverage comparison (main metric = covered count / 102. Stationary and non-stationary reachable patterns number 64/38 respectively. The last three rows are 50% ablation variants). Stage 4-enabled rows report final outputs, whereas No Swap reports the Stage 3 output.

Method / variant	Covered / 102	abs (covered / 128)	Non-stationary (X/38)
Random-25	89 (87.3%)	69.5%	25/38
Random-50	97 (95.1%)	75.8%	33/38
Stratified-25	97 (95.1%)	75.8%	33/38
Stratified-50	99 (97.1%)	77.3%	35/38
BLAST-style-25	92 (90.2%)	71.9%	28/38
BLAST-style-50	99 (97.1%)	77.3%	35/38
TRACS-25	**102 (100%)**	79.7%	**38/38**
TRACS-50	**102 (100%)**	79.7%	**38/38**
Full Corpus	102 (100%)	79.7%	38/38
In-cluster random	102 (100%)	79.7%	38/38
Proportional Quota	102 (100%)	79.7%	38/38
No Swap	93 (91.2%)	72.7%	29/38

Table 15: Paired bootstrap for the ablation variants (each ablation variant vs the TRACS-50 control, MASE; the diff is the arithmetic mean of per-task differences, not the geometric-mean difference in Table 3).

Comparison	MASE diff (control–ablation)	95% CI	p	Win/Tie/Loss
TRACS K=1000	-0.0569	$[-0.1974, +0.0389]$	0.4174	11/0/9
TRACS K=3000	-0.0505	$[-0.2842, +0.1126]$	0.7420	12/0/8
TRACS K=500	-0.0390	$[-0.1743, +0.0739]$	0.5826	15/0/5
TRACS No Swap (Stage 3 only)	-0.0671	$[-0.2770, +0.0740]$	0.5694	12/0/8
TRACS Proportional quota (no rebalancing)	-0.0775	$[-0.2103, +0.0164]$	0.1360	16/0/4
TRACS In-cluster random (no Sobol)	-0.0584	$[-0.1812, +0.0203]$	0.2842	12/0/8

Table 16: Per-task MASE details for the ablation variants (mean over 5 seeds. TRACS-50 is the control, and the remaining columns are ablation variants).

Task	TRACS-50	K=1000	K=3000	K=500	No Swap	Prop. quota	In-cluster random
electricity_15t	0.5993	0.6182	0.6150	0.6052	0.6002	0.6194	**0.5969**
electricity_h	0.8337	0.8304	**0.7846**	0.8376	0.7928	0.8373	0.7947
hierarchical_sales_w	0.8764	0.9135	0.9002	0.8734	**0.8664**	0.8852	0.8711
hospital	1.0236	1.0740	1.0498	1.0188	1.0570	1.0121	**0.9794**
loop_seattle_5t	**0.6502**	0.6655	0.6576	0.6521	0.6559	0.6648	0.6558
loop_seattle_d	0.9440	**0.8804**	0.9390	0.9764	0.8868	0.9106	1.0126
loop_seattle_h	0.7278	0.7153	**0.6549**	0.7360	0.7883	0.7746	0.7251
m4_daily	**2.2028**	2.4304	2.3069	2.2732	2.2221	2.4452	2.2601
m4_hourly	**0.2923**	0.3219	0.2987	0.2945	0.3073	0.3305	0.3069
m4_monthly	**1.2516**	1.2950	1.3940	1.2838	1.4251	1.3432	1.4405
m4_quarterly	**1.4638**	2.6761	3.4256	2.5192	3.2420	2.5999	2.5466
m4_weekly	**1.2408**	1.2841	1.2502	1.2853	1.2449	1.2585	1.2969
m4_yearly	2.5786	2.3635	2.1564	**2.0552**	2.1467	2.2328	2.3652
m_dense_d	0.7745	0.7663	0.7900	0.7669	**0.7541**	0.7553	0.8179
m_dense_h	0.6145	0.6105	**0.5700**	0.6256	0.6064	0.6294	0.6149
solar_10t	4.4998	4.2255	**3.9152**	4.1570	4.1386	4.5882	4.4493
solar_d	0.9669	0.9512	0.9578	0.9750	0.9583	0.9714	**0.9501**
solar_h	3.3980	3.4509	**3.2805**	3.7741	3.5796	3.6221	3.4105
sz_taxi_15t	0.7778	**0.7752**	0.7791	0.7781	0.7796	0.7808	0.7801
sz_taxi_h	**0.6417**	0.6491	0.6425	0.6500	0.6476	0.6466	0.6518

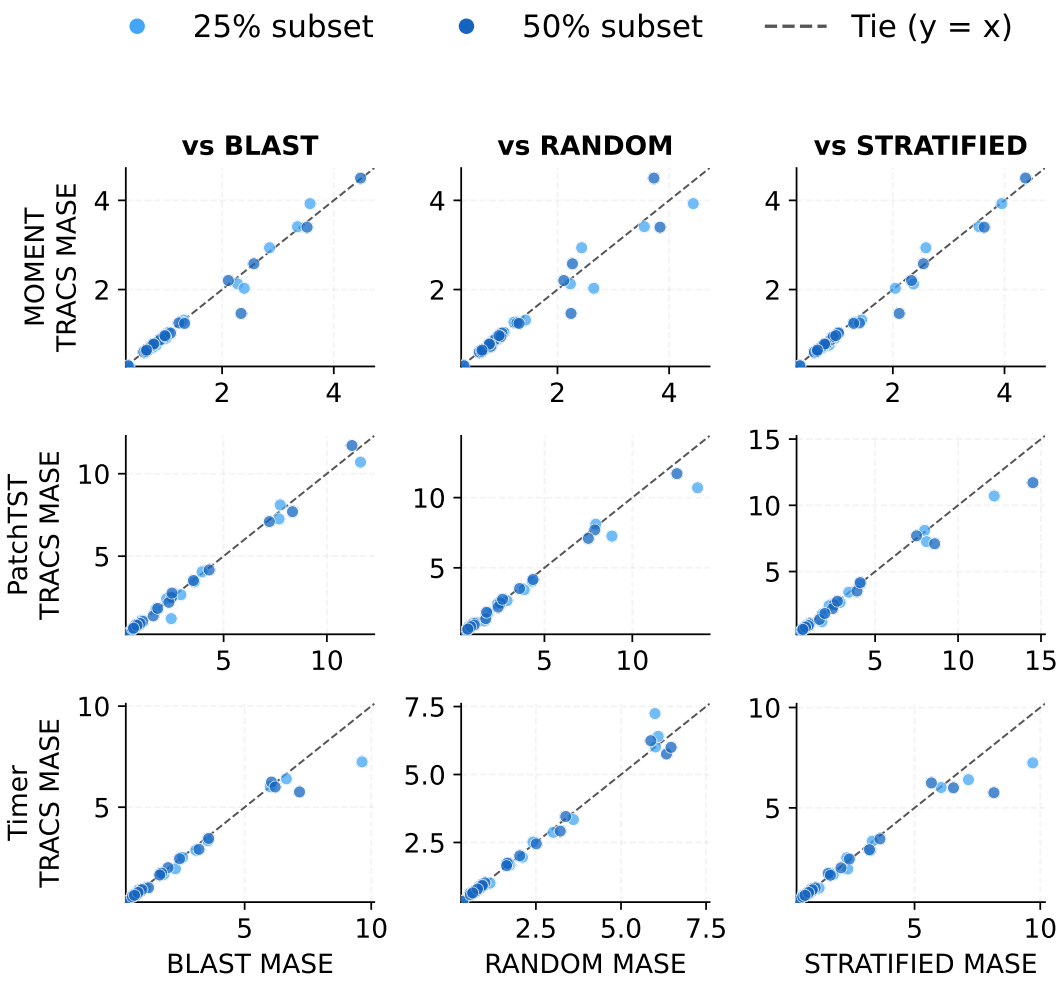


Figure 7: Head-to-head MASE of TRACS against each baseline across tasks. Points below the diagonal mark tasks on which TRACS attains the lower MASE. BLAST denotes BLAST-style.