TSROUTER: TRAINING-FREE CAPABILITY LANDSCAPE DISCOVERY AND ROUTING FOR GROWING TIME-SERIES FOUNDATION MODEL ZOOS

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series foundation models (TSFMs) offer zero-shot forecasts, but their strengths vary across forecasting tasks and change as new models appear. This creates a two-sided selection problem: forwarding every candidate can improve task-specific selection but makes inference expensive as the model zoo grows, whereas supervised selectors avoid online forwarding but require candidate-dependent label updates and retraining after model arrivals. A practical router must retain competitive quality while keeping both inference and adaptation costs bounded. How can a router reuse capability evidence without training a selector for each evolving model zoo? We hypothesize that model preference is locally transferable: forecasting windows with similar temporal structure tend to favor similar models. We introduce TSRouter, a training-free framework for discovering, retrieving, and updating a capability landscape. Representative, source-stratified probe anchors identify model-specific capability regions. A shared representation lets unseen task windows retrieve this evidence, and window/channel rank fusion turns local preferences into a task-level choice. New models are incorporated into the same capability space using existing anchors, preserving previously discovered evidence. On 97 forecasting configurations from 23 datasets and a 20-model zoo, TSRouter achieves the best mean absolute scaled error among the evaluated deployable selectors. Holdout diagnostics and ablations support local capability transfer and evidence aggregation, while release-ordered evaluation demonstrates adaptation to changing candidates. A runtime-aware variant provides a complementary accuracy–latency trade-off.

1 INTRODUCTION

Time-series foundation models (TSFMs) such as Chronos, TimesFM, and Moirai make pretrained zero-shot forecasting available across tasks (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024). As more TSFMs are released, users face a growing choice of forecasters for each new task. Choosing among them is difficult because model strengths vary with frequency, horizon, seasonality, and domain. This variation means that the newest or strongest average model need not be best for a particular task; older models can remain useful in regimes where newer ones are weaker. As Figure 1(a) shows, the zoo therefore grows by adding complementary capabilities, with useful coverage distributed across models. That complementarity makes routing valuable, but also means the candidate pool keeps changing.

A router could evaluate every candidate on each task, but forwarding costs grow with the zoo. A supervised selector avoids that online work, yet its candidate-dependent targets must be updated when models arrive. Figure 1(b) summarizes this quality–efficiency tension. The challenge is to retain competitive selection quality while controlling both inference and adaptation costs, which calls for evidence reusable across tasks and zoo updates.

Existing transferability-estimation and model-selection methods infer task-to-model suitability from labeled task evidence, representations, or candidate evaluations (Nguyen et al., 2020; You et al., 2021; Zhang et al., 2023; Yao et al., 2023; Abdallah et al., 2022). Most assume a fixed candidate set. For a

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.

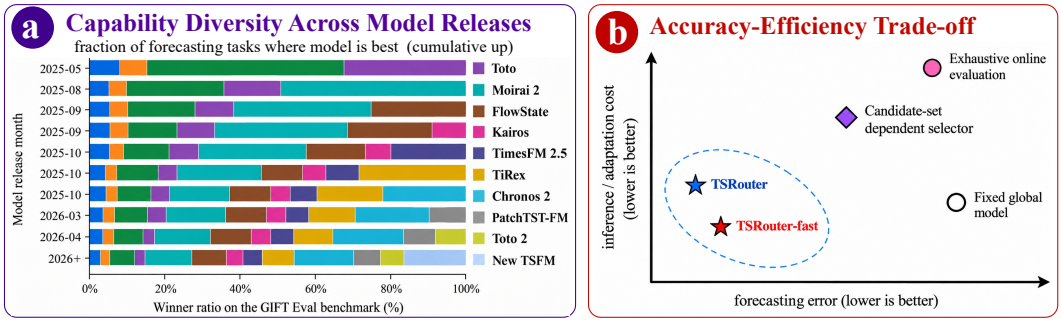


Figure 1: Why a growing TSFM zoo needs an efficient router. The left panel shows complementary capability coverage across release stages: new models add useful regions while earlier models remain best-performing on other tasks. The right panel summarizes the resulting accuracy–efficiency pressure; full-zoo forwarding and selector retraining incur high inference or adaptation costs, while TSRouter reuses fixed capability evidence and targets a competitive operating point.

growing zoo, the open question is whether model strengths can instead be stored as reusable evidence that unseen tasks can retrieve and new models can join without relearning the task-to-model mapping.

We study this question through a capability landscape: representation-space regions supported by measured model performance on fixed probe windows. Our hypothesis is that windows with similar temporal structure tend to favor similar models, allowing local evidence to transfer. Unlike conventional nearest-neighbor retrieval, anchors are evidence about relative model competence, not examples used to construct forecasts. Because individual windows are noisy and similarity does not guarantee shared forecasting difficulty, the router must build representative evidence and aggregate it across the observed task.

TSRouter implements this construction in three stages. PROFILE measures model performance on representative anchors and identifies model-specific capability regions. ROUTE retrieves and aggregates local evidence to rank models for an unseen task. INSERT evaluates a newly released model on the existing anchors and updates the same capability landscape, without relearning the representation or retraining a selector. The default method trains neither a forecaster nor a supervised selector; measured probe losses remain its offline evidence.

We evaluate whether this evidence improves selection, supports local capability transfer, and remains useful as the zoo grows. On the evaluated benchmarks, TSRouter improves aggregate routing quality over static, supervised, representation-based, and online-probing alternatives. Profile-holdout diagnostics and ablations examine locality, representation, anchor coverage, and aggregation; release-ordered evaluation tests adaptation to new candidates. Our contributions are:

- We formulate zero-shot routing in a growing TSFM zoo as the discovery and reuse of model-capability evidence from observed context.
- We introduce a training-free capability landscape that combines representative anchors, model-specific capability regions, and task-level evidence aggregation, with incremental updates for new models.
- We show that this construction supports accurate selection, interpretable local competence, and adaptation to model arrivals on a heterogeneous TSFM evaluation, with a controllable accuracy–latency trade-off.

2 Problem Setting and Method Overview

2.1 Notation and Problem Setting

TSRouter is a training-free routing framework over a growing pool of pretrained time-series foundation models (TSFMs). At release time t, the model pool contains $Z_t = \{m_1, \ldots, m_{M_t}\}$. A task $q = (X_q, H_q, \mathcal{B}_q)$ provides an observed context, a forecast horizon, and optional interface-compatibility constraints. The system must return one enabled model before the future labels of q are

observed. A probe anchor is a real historical context–target example used offline; a task window is sampled only from X_q at routing time. The full symbol table is given in Appendix A.

For an evaluation metric ℓ, let $\ell(m, q)$ denote the loss of model m on the held-out future and let

$$m_q^* = \arg \min_{m \in Z_t,\, m \models \mathcal{B}_q} \ell(m, q) \tag{1}$$

be the evaluation oracle. The oracle is used only after routing for analysis. ROUTE selects $\hat{m}_q$ from the current model pool using observed context and maintained capability evidence; it does not evaluate candidate TSFMs on the task horizon and does not ensemble their forecasts. Zero-shot therefore refers to the unseen task, while offline probes may use their own targets to measure model capability.

2.2 Growing-Zoo Routing Problem

The routing objective is task-adaptive zero-shot selection over a growing zoo. When a new model arrives, $Z_{t+1} = Z_t \cup \{m_{\text{new}}\}$, the best model for a task type can change while older models remain useful elsewhere. A static model-selection policy can become stale, and a selector trained for Z_t must generally update its candidate-dependent targets for Z_{t+1}. TSRouter instead maintains a metric-conditional capability landscape: MASE defines MASE local-best regions, while a probabilistic setting can build a CRPS capability representation from the same measured capability map.

This claim has a precise boundary. TSRouter is a routing framework for existing TSFMs rather than a new forecaster. It targets average routing loss and adaptation cost under a no-online-candidate-evaluation constraint; it does not promise the oracle model on every task. Fixed-zoo quality is one evaluation dimension, while the method objective is to reuse evidence after model arrival without re-profiling old models, replaying historical tasks, or training a new selector.

2.3 Reusable Capability Evidence

TSRouter represents model capability with fixed representative probe anchors. Each anchor is embedded in a shared representation space and associated with measured forecasting losses from the current model zoo. These measurements define model-specific capability regions that can be reused for unseen tasks. The model pool and metric-specific evidence map retain the metadata, compatibility constraints, probe losses, and complete error matrix needed for reproducibility; the full state is detailed in Appendix B.

TSRouter consists of three stages. PROFILE evaluates the current model pool on the fixed probe workload and constructs reusable capability evidence. ROUTE embeds sampled observed windows, measures distances to each model's capability-support set, applies any compatibility constraint, fuses the evidence, and returns one enabled TSFM. INSERT evaluates a newly released model on the same anchors and updates only affected capability regions. Figure 2 gives the dataflow; implementation-level state and accounting are deferred to Appendix B.

2.4 Cost Boundary and Adaptation Model

Let M_t be the zoo size, n the number of anchors, s the number of sampled task windows, and U the number of future tasks over which profiling is amortized. Exhaustive online evaluation pays M_t full forwards per task, and Task-Probe pays sM_t sample forwards. A supervised selector can avoid online candidate evaluation but must evaluate a new model on a labeled pool, update candidate-dependent targets, and retrain or recalibrate after model arrival. TSRouter shifts candidate execution to PROFILE and INSERT: ROUTE uses the fixed representation and retrieved capability regions, while adaptation costs n forwards for the incoming model plus affected-region updates. The selected model's final forecast is excluded from this comparison; formal asymptotics are in Appendix B.

A compact asymptotic comparison is provided in Appendix B.3, Table 5. The main-text accounting boundary is simple: exhaustive online evaluation scales with the candidate pool, supervised selectors require candidate-dependent target updates after model arrival, and TSRouter evaluates candidates only during PROFILE/INSERT while reusing the fixed capability representation during ROUTE.

3 TRAINING-FREE CAPABILITY DISCOVERY AND ROUTING

TSRouter uses the same representation for probe anchors and observed task windows. Figure 2 shows how this shared space connects capability discovery, retrieval, and update.

3.1 PROFILE: DISCOVERING CAPABILITY REGIONS

Let $D_p = \{a_i\}_{i=1}^n$ be a fixed set of probe anchors and ψ a fixed representation map. To cover heterogeneous temporal patterns, we stratify candidate windows by source, cluster within each source in representation space, and choose the real window nearest each centroid. The default uses $n = 3000$ anchors from public pretraining sources associated with Chronos, Moirai, and TimesFM. These probes are separate from evaluation tasks. The Fourier representation combines seeded projections, fast Fourier transform (FFT) magnitudes, multi-scale difference energies, and statistics; its weights remain fixed. Appendix B gives the construction details.

Capability hypothesis. Our central hypothesis is that forecasting windows with related temporal structure occupy nearby points in a shared representation space, and that nearby windows are more likely to favor similar models than unrelated windows. ROUTE aggregates such local window-level evidence to make a task-level decision. A fixed representation is useful only if this locality survives the change from an offline probe anchor to an unseen task window; it need not preserve a global model ranking. We therefore test the hypothesis at the level of local evidence, using profile-holdout recovery, task-level local best model recovery, and the independent heatmap rather than treating similarity as a guarantee.

The intended mechanism has three parts. First, a *shared representation* places probe anchors and observed task windows in one coordinate system, so a task can retrieve evidence without executing every candidate. Second, each model has *local competence*: it may be best only for a subset of temporal regimes, even when its average score is weaker. Third, the subset of nearby anchors with a common local best model forms a *capability region*. Source-stratified clustering makes these regions cover multiple domains and frequencies, while retaining real windows keeps the evidence tied to interpretable forecasting behavior. The region is empirical and metric-conditional; it records where a model has measured advantage, not a calibrated probability that it will win every nearby task. This is why the full loss matrix remains available for audit and insertion even though ROUTE uses a compact capability representation.

Why not train a selector? The distinction from a small classifier is selector label dependency. A supervised selector needs labels such as the best model in the current pool or a candidate-relative rank for historical tasks. When a new model arrives, those labels can change even for old tasks, so the selector must evaluate the arrival on a labeled pool and refresh, recalibrate, or retrain its output. TSRouter instead measures each model on the same fixed anchors, keeps the complete evidence matrix, and updates only affected capability regions. The approach is therefore training-free with respect to task labels at ROUTE time; it does not claim that a learned selector is impossible, only that label adaptation is a separate cost in a growing model pool. Appendix B gives the compatibility, leakage, latency, and complexity details behind this boundary.

PROFILE is the offline stage that converts the model zoo into reusable capability evidence. Window-level evidence transfers across downstream tasks because both probe windows and task windows are embedded in the same space; channel- or task-level labels would be tied to a particular downstream dataset. The probe split, source manifest, and anchor provenance are stored as artifact metadata, so this boundary can be audited independently of the routing benchmark.

The encoder observes only the context portion of each anchor. For each anchor, we store $z_i = \psi(a_i)$ and measure every model's probe loss $e^{\ell}_{m,i}$. The metric-specific local best model and its region are

$$w_i^{\ell} = \arg\min_{m \in Z_t} e^{\ell}_{m,i}, \qquad A_t^{\ell}(m) = \{a_i \in D_p : w_i^{\ell} = m\}. \tag{2}$$

The capability map P_t^{ℓ} retains the full loss matrix; the capability representation I_t^{ℓ} associates representations with the local-best partition. A region represents empirical local advantage, not a learned probability of success. Source stratification improves coverage, while real anchors retain interpretable

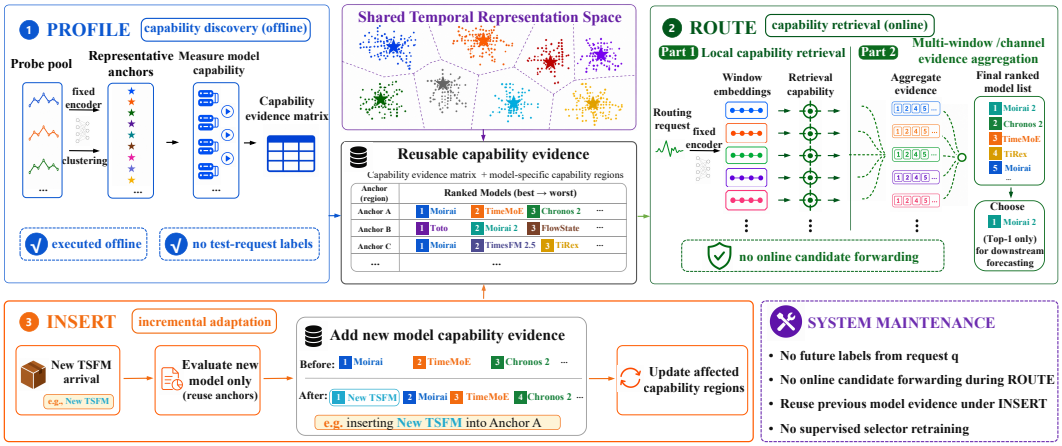


Figure 2: TSRouter overview and information boundary. PROFILE evaluates the model pool on representative probe anchors and identifies model-specific capability regions; ROUTE embeds only observed task windows, retrieves local evidence, and fuses window/channel ranks to return one model; INSERT evaluates only the arriving model on the same anchors and updates affected regions. Future targets enter only after selection for evaluation.

forecasting evidence. The local-best partition compresses this evidence for retrieval; near-optimal models remain recorded in the loss matrix but do not define that anchor's primary region.

The full error matrix is retained because INSERT and audit checks need more than the current region assignment. The local-best partition is the retrieval compression: a model is represented by the subset of anchors on which it empirically performs best, rather than by one global descriptor. The resulting region can be sparse for a specialized model and dense for a broadly useful model, but both remain searchable under the same coordinate system.

3.2 ROUTE: Aggregating Local Capability Evidence

ROUTE samples windows from observed context X_q, applies the fixed representation pipeline ψ, and ranks models. Candidate TSFMs are not executed; only the selected model runs afterward. Core-ROUTE measures method-specific comparison after sampling, ROUTE also includes sampling and preprocessing, and end-to-end latency further includes the selected TSFM's forecast.

For a window x drawn from observed context, the evidence for model m is its nearest supporting anchor:

$$d_x^\ell(m) = \min_{a_i \in A_t^\ell(m)} \mathrm{dist}(\psi(x), z_i). \tag{3}$$

An empty region receives infinite distance. This compares each model's closest evidence, rather than assigning the task to a single nearest anchor. Let $r_x^\ell(m)$ be the rank induced by these distances. For channel c, let $\mathcal{W}_{q,c}$ denote sampled observed windows and C_q the task's channel count. Minimum-rank reduction within channels and mean fusion across channels give

$$r_{q,c}^\ell(m) = \min_{x \in \mathcal{W}_{q,c}} r_x^\ell(m), \qquad r_q^\ell(m) = \frac{1}{C_q} \sum_{c=1}^{C_q} r_{q,c}^\ell(m). \tag{4}$$

TSRouter-main selects $\hat{m}_q = \arg\min_{m \in Z_t,\, m \models \mathcal{B}_q} r_q^\ell(m)$, using 20 sampled windows by default. Rank fusion combines locally favorable evidence without assuming raw distances are calibrated prediction errors. Its effectiveness is tested by the window and representation ablations.

TSRouter-fast adds a stored runtime prior. With profile-time runtime τ_m normalized to $\omega_m^{\mathrm{fast}} \in [0, 1]$, it minimizes $s_q^\ell(m) = r_q^\ell(m) + \lambda \omega_m^{\mathrm{fast}}$, where $\lambda = 0.5$. Both modes consult recorded evidence; only the selected TSFM executes afterward. The prior changes the accuracy–latency operating point without changing the discovery mechanism.

The optional compatibility mask is an inference-boundary check rather than a quality heuristic: it may enforce horizon, context-length, variate, probabilistic-output, hardware, or application-setting constraints before the final argmin. The benchmark uses a normalized interface in which all models in the zoo are enabled for the reported MASE tasks, so the main comparison isolates capability evidence. TSRouter-fast reads stored runtime logs rather than executing candidates during ROUTE.

3.3 INSERT: Updating an Evolving Landscape

The fixed anchor space makes model arrival an evidence update. INSERT evaluates m_{new} on D_p, appends its losses to P_t^ℓ, and compares them with the current local-best assignments. Anchors for which the incoming model is locally preferable change regions; other assignments and all old-model measurements remain reusable. The updated capability representation I_{t+1}^ℓ therefore expands the candidate label space over a fixed evidence space. No historical routing task is replayed and no selector is retrained.

This update distinguishes expanding the candidate set from relearning a fixed output space. It still incurs new-model inference on the anchors and capability-region updates, but preserves the coordinate system in which older capability evidence was obtained. The release-ordered experiment tests the resulting selection behavior; implementation details are in Appendix B.

Operationally, INSERT is a per-anchor comparison update. The incoming model contributes one new error value per anchor; only anchors whose local best model changes move between capability-support subsets. Existing model-profile columns are not recomputed and historical tasks are not replayed. This is the distinction from selector retraining, which must usually evaluate the new model on a labeled pool, update candidate-dependent targets, and retrain or recalibrate the output head.

4 Experiments

We test three linked questions: whether capability evidence improves selection, whether representation-space locality supports that transfer, and whether the evidence remains useful as the zoo expands.

Setup. TSFM-ZooBench covers 97 dataset–frequency–horizon configurations from 23 GIFT-Eval datasets (Aksu et al., 2024) and 20 TSFMs across 12 families. Tasks use at most 512 context points, with all candidates enabled under normalized MASE. Profile anchors are separate from evaluation tasks; future targets are used only after routing. Baselines span static Random/Recent and Profile-probe, supervised AutoForecast (Abdallah et al., 2022), resource-aware AutoXPCR (Fischer & Saadallah, 2024), a SimpleTS-style representation classifier (Yao et al., 2023), and Task-probe, the only method that evaluates candidates on observed task windows. Current-best is a non-deployable static oracle selected with test labels. Appendices C–D detail the protocol and model pool.

Metrics. Mean absolute scaled error (MASE) is the primary loss. Regret is selected-model MASE minus per-task oracle MASE, averaged across tasks; Rank is the selected model's mean true MASE rank. Task WR@k is the fraction where the selected Top-1 model belongs to the true Top-k, not a Top-k ensemble. Continuous ranked probability score (CRPS) is auxiliary. Core-ROUTE measures post-sampling comparison; ROUTE includes sampling/preprocessing, and end-to-end (E2E) adds the chosen model's forecast. Online costs use P95; INSERT reports mean seconds on one RTX 3090.

Model pool. Table 1 defines the family and size abbreviations used by the results and reports representative baseline performance; all experiments use the complete 20-model pool.

4.1 Does Capability Evidence Improve Model Selection?

Table 2 shows that task-adaptive selection improves on choosing one model for every task: TSRouter-main reaches MASE 1.508, compared with 1.570 for the retrospective best single model and 1.558 for the strongest non-TSRouter deployable baseline. It also has the lowest deployable regret (0.104) and mean rank (4.619), and the highest Task WR@1/3 (0.268/0.567). These measures agree that its gains extend beyond a few exact wins: selecting both Chronos 2 and TiRex lets it exploit complementary

Table 1: Representative TSFM families and baseline performance in the 20-model pool. One representative variant per family is shown for readability, while all experiments use the complete 20-model pool. Capability coverage is the fraction of labeled routing tasks for which the static model is the MASE local best model and is summarized by the left panel of Figure 1. Lower MASE, Rank, CRPS, and E2E P95 are better; release months refer to the evaluated model-pool snapshot.

TSFM family	Size variants	Release month	Rep. model	MASE ↓	Rank ↓	Capability coverage ↑	CRPS ↓	E2E P95(s) ↓
Moirai (Woo et al., 2024)	small (Moi.S), base (Moi.B), large (Moi.L)	2024-03	Moi.L	1.918	14.134	0.031	0.253	34.36
TimeMoE (Maple728, 2024)	50M (TMoE.50)	2024-09	TMoE.50	2.407	15.763	0.031	0.331	250.09
Chronos-Bolt (Amazon Web Services, 2024)	tiny (Chr.bT), mini (Chr.bM), small (Chr.bS), base (Chr.bB)	2024-11	Chr.bB	1.828	8.845	0.052	0.229	240.82
Toto (Cohen et al., 2024)	base-1.0 (Toto)	2025-05	Toto	1.656	11.134	0.031	0.234	514.36
Moirai 2 (Salesforce AI Research, 2025)	small (Moi2.S)	2025-08	Moi2.S	**1.570**	7.567	<u>0.155</u>	<u>0.221</u>	55.44
FlowState (Graf et al., 2026)	r1 (Flo.r1)	2025-09	Flo.r1	1.593	8.701	0.113	0.229	32.00
Kairos (Feng et al., 2025)	10M (Kai.10), 23M (Kai.23), 50M (Kai.50)	2025-09	Kai.50	1.790	9.474	0.021	0.243	26.27
TimesFM 2.5 (Google Research, 2025)	200M PyTorch (TFM.25)	2025-10	TFM.25	1.654	7.649	0.052	0.227	74.01
TiRex (Auer et al., 2025)	base (TiRex 1.1)	2025-10	TiRex	1.627	**5.619**	0.103	**0.217**	219.14
Chronos 2 (Amazon Web Services, 2025)	base (Chr.2)	2025-10	Chr.2	<u>1.573</u>	<u>5.887</u>	**0.186**	0.239	<u>24.78</u>
PatchTST-FM (IBM Research, 2026)	r1 (PTS.FM)	2026-03	PTS.FM	1.651	8.165	0.083	0.227	121.68
Toto 2 (Datadog, 2026)	4M (Toto2.T), 22M (Toto2.S)	2026-04	Toto2.S	1.831	11.649	0.072	0.294	**11.53**

Table 2: Main routing quality and inference latency on the 20-model zoo. Regret is the selected-model MASE gap to the per-task oracle, and Task WR@1/3 tests whether the recommended Top-1 model belongs to the true MASE Top-1/Top-3 set. Selected-model frequency is a selection statistic, not capability coverage. Core-ROUTE P95 is post-sampling comparison cost; E2E P95 also includes selected-model inference. Lower MASE, Regret, Rank, CRPS, and latency are better, while higher Task WR@1/3 is better. Current-best is a non-deployable test-label oracle; bold/underline mark the best/second-best deployable values.

Method group	Method	MASE ↓	Regret ↓	Rank ↓	Task WR@1/3 ↑	CRPS ↓	Selected models frequency (Top/2nd)	Core-ROUTE P95 (s) ↓	E2E P95 (s) ↓
static	Random	1.746	0.342	10.206	0.058/0.159	0.246	Moi2.S 6.8% / Chr.bM 6.4%	–	68.98
static	Recent	1.831	0.427	11.619	0.082/0.175	0.294	Toto2.S 100.0% / –	–	**11.53**
static	Profile-probe	1.573	0.169	5.814	0.186/0.464	0.239	Chr.2 100.0% / –	0.52	40.18
representation	SimpleTS	1.654	0.250	7.546	0.052/0.186	0.227	TFM.25 100.0% / –	3.00	101.90
supervised	AutoForecast	1.590	0.187	5.649	0.103/0.433	<u>0.219</u>	TiRex 63.9% / Chr.2 10.3%	1.11	203.42
resource-aware	AutoXPCR	1.558	0.154	6.165	0.165/0.412	0.238	Chr.2 50.5% / TFM.25 29.9%	1.19	56.21
forward-probe	Task-probe	1.634	0.230	6.237	<u>0.227</u>/0.433	**0.213**	Chr.2 15.5% / TiRex 14.4%	73.10	229.28
ours	TSRouter-main	**1.508**	**0.104**	**4.619**	**0.268/0.567**	0.233	Chr.2 58.8% / TiRex 20.6%	<u>0.48</u>	137.30
ours	TSRouter-fast	<u>1.530</u>	<u>0.126</u>	8.742	0.124/0.289	0.225	Toto2.T 51.5% / Chr.2 26.8%	**0.44**	<u>33.49</u>
static oracle	Current-best	1.570	0.166	7.423	0.155/0.309	0.221	Moi2.S 100.0% / –	not deployable	55.44

strengths across tasks. Its CRPS is not the best, so the evidence supports improved MASE-oriented routing rather than uniform gains on every metric.

Direct online probing does not improve task-level selection here. Task-probe evaluates every candidate on observed windows, but its MASE (1.634) remains above TSRouter-main (1.508). Its Core-ROUTE P95 is 73.10 seconds, versus 0.48 seconds for TSRouter-main, because candidate execution is part of the probe. This comparison supports the value of reusing representative offline evidence when candidate forwarding is costly; it does not show that nearest-neighbor geometry alone suffices or that online probing is ineffective in every setting.

Figure 3 makes this quality–latency choice explicit. TSRouter-fast raises MASE modestly from 1.508 to 1.530, while reducing E2E P95 from 137.30 to 33.49 seconds (about 76%). It remains more accurate than the other deployable baselines, though its mean rank (8.742) is worse than TSRouter-main (4.619), and Recent is still faster. Thus the runtime prior offers a distinct operating point, not a uniformly superior selector.

4.2 Does the Representation Preserve Local Competence?

Table 8 links representation quality to both local evidence and downstream routing. On profile holdout, fixed Fourier features recover local-best assignments with WR@1 $= 0.446$, versus 0.076 for identity; task-level recovery is likewise higher (0.268 versus 0.206). The parallel changes are consistent with the claim that useful representation geometry helps local evidence transfer, while the non-perfect scores leave room for ambiguous windows and imperfect assignments.

Figure 4 tests whether these assignments correspond to relative forecasting competence rather than a global average. Each row groups profile-holdout anchors assigned to one model, and each column reports a candidate model's mean true MASE rank on those anchors; lower diagonal values indicate that region-associated models tend to rank well in their assigned regions. This offers a direct check of locality, separate from task-level selection scores. It does not imply pure regions or universal winners:

neighboring windows can have ambiguous preferences, which motivates aggregating evidence across windows and channels.

The representation controls connect locality to downstream recommendation. Identity weakens region purity, within-region rank, and task recovery; random anchors retain some task-level signal but produce noisier regions. MLP and SimpleTS-TS2Vec recover part of the locality, but not the Fourier configuration's combined region quality and task recovery. Read together, the heatmap tests whether the assigned regions encode local competence, while held-out task metrics test whether that signal helps choose a model. The agreement supports transfer within this protocol, not universal transfer across benchmarks.

4.3 Can Capability Evidence Adapt to Model Arrivals?

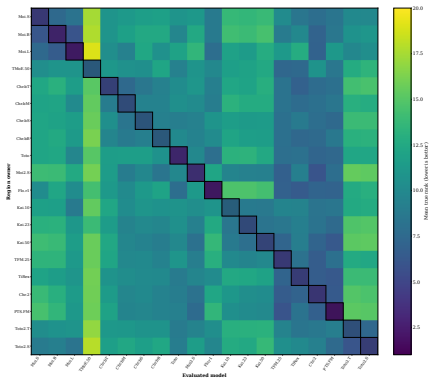


Figure 4: Model-specific capability regions on the profile holdout. Each cell is the mean true MASE rank of the column model on anchors assigned to the row model; lower is better. Black boxes mark region-associated cells, and darker diagonal cells support local competence.

Figure 5 tests whether the same evidence remains useful as candidates arrive. Recent tracks the newest model as a fixed choice and can rise or fall, while TSRouter can retain older capabilities and add evidence from each evaluated arrival. Its improving overall trajectory is consistent with that reuse mechanism, rather than a rule that newer models are always better. The experiment covers the evaluated release stream only; it does not establish transfer to unevaluated models. Appendix E, Figure 6, reports measured update costs separately from the projected 40/80-model stress points.

4.4 Which Components Support Transfer?

To isolate the source of the transfer signal, Table 3 changes one representation, anchor-construction, or evidence-aggregation choice at a time from the fixed default. The comparison separates locality and coverage effects from the task-level aggregation that turns window evidence into a routing decision. Table 3 shows that representation choice affects both quality and cost. Replacing Fourier features with selector-trained SimpleTS-TS2Vec gives the largest MASE deterioration (1.5076 to 1.6023) and the highest Core-ROUTE P95 (3.45 seconds) among these controls. Thus a representation useful for supervised selection is not automatically suited to capability-region retrieval; the trained encoder is a diagnostic replacement, while the default remains training-free.

Table 3: Component ablations of TSRouter on the 20-model zoo. Each row changes one component of the default configuration and reports routing quality, probabilistic quality, Core-ROUTE P95 latency, and mean INSERT adaptation time. The default uses fixed Fourier features, 3,000 source-stratified clustered anchors, 20 windows, and all-channel fusion; the rows isolate representation, source-stratified coverage, anchor budget, and multi-window/channel aggregation.

Block	Ablation	Changed setting	MASE ↓	Task WR@1 ↑	CRPS ↓	Core-ROUTE P95 ↓	INSERT mean (s) ↓
Reference	Full	default	**1.5076**	0.268	0.2335	0.48	97.88
Representation	w/o Enc	Encoder: Fourier → identity	1.5539	0.206	0.2350	0.65	97.88
Representation	w/ MLP	Encoder: Fourier → random MLP	1.5449	0.227	0.2396	0.42	97.88
Representation	w/ SimpleTS- TS2Vec	Encoder: Fourier → SimpleTS-TS2Vec	1.6023	0.186	0.2391	3.45	97.88
Profile	w/o Cluster	Anchors: clustered → random	1.5172	0.247	0.2385	0.60	97.88
Profile	w/o Chronos src.	Profile sources: all → no Chronos	1.5625	0.206	**0.2268**	0.42	97.88
Profile	Fewer anchors	Anchors: 3000 → 1500	1.5519	0.196	0.2335	0.29	**51.67**
Profile	More anchors	Anchors: 3000 → 5000	1.5252	0.227	0.2413	0.60	153.84
Evidence aggregation	10 windows	Route windows: 20 → 10	1.5116	**0.278**	0.2301	0.20	97.88
Evidence aggregation	1 window	Route windows: 20 → 1	1.5535	0.247	0.2353	**0.02**	97.88
Evidence aggregation	w/o ChanFuse	Channels: all → first	1.5094	0.268	0.2337	0.51	97.88

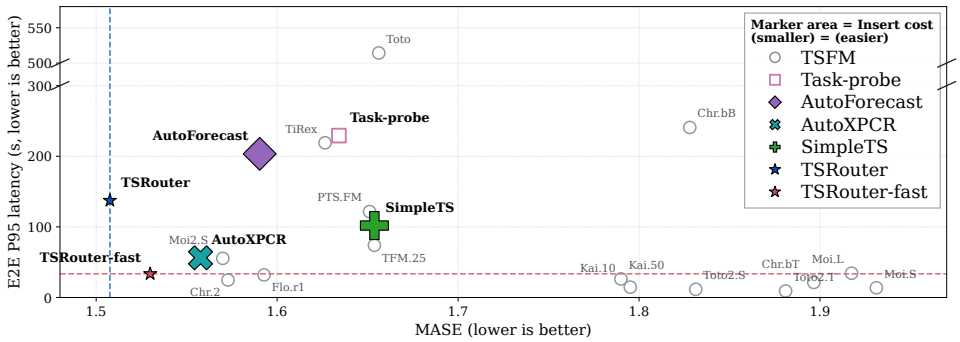


Figure 3: Fixed-zoo accuracy–latency trade-off on the 20-model zoo. The x-axis reports MASE and the y-axis reports E2E P95 forecast latency, so lower-left is better. Single-TSFM points are fixed model choices; selector points include routing and selected-model inference. Bubble size encodes mean INSERT adaptation time for routing methods. TSRouter-fast trades a small MASE increase for substantially lower tail latency.

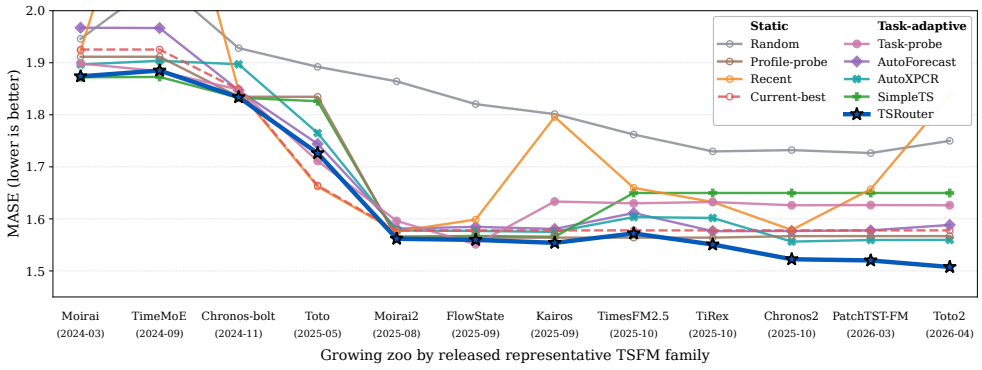


Figure 5: Routing quality under simulated release-order zoo growth. Each x-axis tick adds one representative TSFM family in its evaluated release order, and the y-axis reports MASE after routing within the current stage; lower is better. Recent shows whether the newest model is a good fixed model choice, whereas a stable decreasing TSRouter curve indicates that the router absorbs useful arrivals without being dominated by weaker releases.

Anchor coverage matters beyond the anchor budget itself. Source ablations show that representative coverage contributes to routing quality, while clustered anchors provide more reliable evidence than random sampling. Increasing the anchor budget improves coverage only marginally at a higher insertion cost, revealing a quality–efficiency trade-off. Similarly, multi-window aggregation improves robustness over single-window routing with limited additional overhead, whereas channel fusion has little effect in this setting. These results support representative evidence construction and adaptive aggregation as key components of TSRouter.

5 Conclusion

TSRouter discovers and maintains a reusable capability landscape for zero-shot selection among heterogeneous TSFMs. Fixed anchors connect model profiling to task-time retrieval, rank fusion aggregates local evidence, and incremental capability-region updates accommodate new candidates. The evaluated zoo demonstrates improved selection quality, informative local competence, and adaptation to model arrivals without selector retraining. Capability evidence thus provides a practical basis for routing within the evaluated TSFM zoo and benchmark setting.

Reproducibility Statement

We provide an anonymous supplementary ZIP containing the implementation of TSRouter, the TSFM-ZooBench evaluation workflow, baseline implementations, configuration files, and scripts used for the experiments. The supplementary package includes instructions for obtaining the required benchmark data and upstream model checkpoints from their original sources. The main operations of TSRouter, including capability profiling, task-level routing, model insertion, and result generation, are implemented in the submitted source code.

AI Use Statement

Generative AI tools were used for language editing, polishing the writing, and LaTeX formatting. They were not used to generate research ideas, experimental code, data, or images. The authors take responsibility for the final manuscript.

Ethics Statement

The study routes publicly available pretrained models on public forecasting benchmarks. Application settings should respect dataset and model licenses, protect input-series privacy, and validate forecast quality before decisions with significant consequences. The reported findings concern a 20-model zoo and its probe coverage. Best-performing-model-only regions omit near-optimal alternatives, representation similarity does not guarantee transfer, and hardware-specific runtimes may change the preferred operating point. Public-model pretraining contamination cannot be ruled out, and predicted zoo-size costs are distinct from measured quality. Further scope is discussed in Appendix G.

References

Mustafa Abdallah, Ryan A. Rossi, Kanak Mahadik, Sungchul Kim, Handong Zhao, and Saurabh Bagchi. AutoForecast: Automatic Time-Series Forecasting Model Selection. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, pp. 5–14, 2022. doi: 10.1145/3511808.3557241. URL https://doi.org/10.1145/3511808.3557241.

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C. Fowlkes, Stefano Soatto, and Pietro Perona. Task2Vec: Task Embedding for Meta-Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6430–6439, 2019. URL https://openaccess.thecvf.com/content_ICCV_2019/html/Achille_Task2Vec_Task_Embedding_for_Meta-Learning_ICCV_2019_paper.html.

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. *arXiv preprint arXiv:2410.10393*, 2024. URL https://arxiv.org/abs/2410.10393.

Amazon Web Services. Chronos-Bolt Base Model Card. Hugging Face model card, 2024. URL https://huggingface.co/amazon/chronos-bolt-base. Accessed September 25, 2026.

Amazon Web Services. Chronos 2 Model Card. Hugging Face model card, 2025. URL https://huggingface.co/amazon/chronos-2. Accessed September 25, 2026.

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

Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. TiRex: Zero-Shot Forecasting across Long and Short Horizons with Enhanced

In-Context Learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/5356603f9c47399adfd372f77a677057-Abstract-Conference.html.

Ben Cohen, Emaad Khwaja, Kan Wang, Charles Masson, Elise Ramé, Youssef Doubli, and Othmane Abou-Amal. Toto: Time Series Optimized Transformer for Observability. *arXiv preprint arXiv:2407.07874*, 2024. URL https://arxiv.org/abs/2407.07874.

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*, pp. 10148–10167, 2024. URL https://proceedings.mlr.press/v235/das24c.html.

Datadog. Toto 2.0 22M Model Card. Hugging Face model card, 2026. URL https://huggingface.co/Datadog/Toto-2.0-22m. Accessed September 25, 2026.

Kun Feng, Shaocheng Lan, Yuchen Fang, Wenchao He, Sihan Lu, Shuqi Gu, Lintao Ma, Xingyu Lu, and Kan Ren. Kairos: Toward Adaptive and Parameter-Efficient Time Series Foundation Models. *arXiv preprint arXiv:2509.25826*, 2025. URL https://arxiv.org/abs/2509.25826.

Raphael Fischer and Amal Saadallah. AutoXPCR: Automated Multi-Objective Model Selection for Time Series Forecasting. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 806–815, 2024. doi: 10.1145/3637528.3672057. URL https://doi.org/10.1145/3637528.3672057.

Google Research. TimesFM 2.5 200M PyTorch Model Card. Hugging Face model card, 2025. URL https://huggingface.co/google/timesfm-2.5-200m-pytorch. Accessed September 25, 2026.

Lars Graf, Thomas Ortner, Stanisław Woźniak, and Angeliki Pantazi. FlowState: Sampling-Rate-Equivariant Time-Series Forecasting. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306, 2026. URL https://arxiv.org/abs/2508.05287v3.

IBM Research. PatchTST-FM R1 Model Card. Hugging Face model card, 2026. URL https://huggingface.co/ibm-research/patchtst-fm-r1. Accessed September 25, 2026.

Ziyu Li, Hilco van der Wilk, Danning Zhan, Megha Khosla, Alessandro Bozzon, and Rihan Hai. Model Selection with Model Zoo via Graph Learning. In *Proceedings of the IEEE International Conference on Data Engineering*, pp. 1296–1309, 2024. doi: 10.1109/ICDE60146.2024.00088. URL https://doi.org/10.1109/ICDE60146.2024.00088.

Maple728. TimeMoE-50M Model Card. Hugging Face model card, 2024. URL https://huggingface.co/Maple728/TimeMoE-50M. Accessed September 25, 2026.

Cuong Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. LEEP: A New Measure to Evaluate Transferability of Learned Representations. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 7294–7305, 2020. URL https://proceedings.mlr.press/v119/nguyen20b.html.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods. *Proceedings of the VLDB Endowment*, 17 (9):2363–2377, 2024. doi: 10.14778/3665844.3665863. URL https://arxiv.org/abs/2403.20150.

Xiangfei Qiu, Xiuwen Li, Ruiyang Pang, Zhicheng Pan, Xingjian Wu, Liu Yang, Jilin Hu, Yang Shu, Xuesong Lu, Chengcheng Yang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, and Bin Yang. EasyTime: Time Series Forecasting Made Easy. In *Proceedings of the 41st IEEE International Conference on Data Engineering*, pp. 4564–4567, 2025. doi: 10.1109/ICDE65448.2025.00353. URL https://arxiv.org/abs/2412.17603.

Salesforce AI Research. Moirai 2.0 R-Small Model Card. Hugging Face model card, 2025. URL https://huggingface.co/Salesforce/moirai-2.0-R-small. Accessed September 25, 2026.

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*, pp. 53140–53164, 2024. URL https://proceedings.mlr.press/v235/woo24a.html.

Yuanyuan Yao, Dimeng Li, Hailiang Jie, Lu Chen, Tianyi Li, Jie Chen, Jiaqi Wang, Feifei Li, and Yunjun Gao. SimpleTS: An Efficient and Universal Model Selection Framework for Time Series Forecasting. *Proceedings of the VLDB Endowment*, 16(12):3741–3753, 2023. doi: 10.14778/3611540.3611561. URL https://www.vldb.org/pvldb/vol16/p3741-chen.pdf.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. LogME: Practical Assessment of Pre-trained Models for Transfer Learning. In *Proceedings of the 38th International Conference on Machine Learning*, 2021. URL https://proceedings.mlr.press/v139/you21b.html.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. TS2Vec: Towards Universal Representation of Time Series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022. doi: 10.1609/aaai.v36i8.20881. URL https://ojs.aaai.org/index.php/AAAI/article/view/20881.

Yi-Kai Zhang, Ting-Ji Huang, Yao-Xiang Ding, De-Chuan Zhan, and Han-Jia Ye. Model Spider: Learning to Rank Pre-Trained Models Efficiently. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/2c71b14637802ed08eaa3cf50342b2b9-Abstract-Conference.html.

APPENDIX READER NAVIGATION

Appendix A defines notation. Appendix B supplements the three method operations. Appendix C specifies data visibility and evaluation, and Appendix D documents candidate models and baseline access. Appendix E gives detailed local-competence and runtime evidence supporting Section 4. Appendix F positions the approach, and Appendix G discusses scope and future directions.

A NOTATION

Table 4 distinguishes task-time observations, offline anchors, metric-specific evidence, and update state. The metric superscript ℓ always identifies the forecasting loss used to assign capability regions. A model ranking is a routing decision rule, not an estimate of that loss or a calibrated confidence.

Table 4: Notation for TSRouter's growing-zoo routing, profiling, and insertion model. The notation separates task-time objects, fixed probe anchors, metric-specific profile and capability-representation state, runtime statistics, and affected records updated during INSERT.

Symbol	Meaning
t, Z_t, M_t	Release time, current TSFM zoo, and zoo size.
ℓ	Evaluation metric and capability-representation objective, such as MASE or CRPS.
m, m_{new}	TSFM in the model pool and newly arriving TSFM.
$q = (X_q, H_q, \mathcal{B}_q)$	Forecasting task with context, horizon, and optional compatibility constraints.
X_q, Y_q	Observed context and held-out future labels for evaluation.
$D_p = \{a_i\}_{i=1}^n$	Fixed probe workload of n representative anchors.
$\psi(\cdot)$, $z_i = \psi(a_i)$, d	Representation encoder, anchor embedding, and dimension.
$e_{m,i}^\ell$, w_i^ℓ	Metric-ℓ probe error of model m on anchor a_i, and the assigned local best model.
$A_t^\ell(m)$	Metric-ℓ local-best anchor subset assigned to model m under zoo Z_t.
P_t^ℓ, I_t^ℓ	Metric-specific capability map and capability representation at release time t.
s, C_q	Number of sampled task windows and number of channels in task q.
$d_x^\ell(m)$, $r_q^\ell(m)$, $\hat{m}_q$	Window-to-model distance, fused task rank, and selected model.
$F_q(m)$, $\bar{F}_q(m)$, $F_p(m)$, $F_s(m)$	Cost on one task, labeled-pool average, probe anchor, and sampled window.
N_{lab}, G_q, E_ψ	Labeled tasks, meta-feature cost, and encoding cost.
Δ_t, τ_m, ω_m^{fast}	Affected capability-representation records, runtime statistic, and efficiency prior.
$\mathcal{W}_{q,c}$, r_x^ℓ, $r_{q,c}^\ell$	Observed windows in channel c, window rank, and channel rank.
s_q^ℓ, λ, m_q^*	Runtime-adjusted score, prior weight, and evaluation oracle.
$\bar{F}_s$, C_{train}	Average sampled-window forward cost and selector training cost.

B METHOD DETAILS AND OPERATION CONTRACTS

B.1 REPRESENTATION AND ANCHOR CONSTRUCTION

TSRouter first maps candidate probe windows into a fixed representation space using ψ. The default implementation uses a fixed Fourier feature encoder: seeded Fourier projections are combined with low-frequency FFT magnitudes, multi-scale difference energies, and statistical features, followed by standard scaling. The projection weights are materialized once and then kept fixed, so Fourier is an encoder configuration rather than a random routing policy. Rather than clustering a single mixed pool, TSRouter stratifies the probe corpus by source and clusters each source separately. Within each source stratum, it runs KMeans in the representation space and materializes each cluster by selecting the real window nearest to the corresponding centroid. The union of these nearest-real-window representatives forms a compact anchor set of 3000 windows.

This source-stratified construction has two purposes. First, it prevents large or redundant pretraining-domain sources from dominating the anchor budget. Second, it preserves an auditable link from each anchor back to a real time-series window rather than using synthetic centroids that no TSFM can directly forecast. The clustering step is only a coverage mechanism: it does not learn a supervised selector or assign task labels. Instead, it builds compact representative anchors whose neighborhoods later support capability-representation retrieval with transferable model-competence evidence, not task-specific labels or online model trials.

B.2 Capability State and Update Semantics

PROFILE(Z_t, D_p, ℓ)
Reads: model pool, fixed probe workload.
Writes: P_t^ℓ, I_t^ℓ.

1. Embed anchors: $z_i \leftarrow \psi(a_i)$.
2. For each $m \in Z_t$, run m on anchors.
3. Store $e_{m,i}^\ell$ and compatibility mask.
4. Assign $w_i^\ell \leftarrow \arg\min_m e_{m,i}^\ell$.
5. Materialize capability map P_t^ℓ and capability representation I_t^ℓ.

ROUTE($q, I_t^\ell, \mathcal{B}_q$)
Reads: capability-representation snapshot, observed context.
Returns: selected model id.

1. Read the current capability representation.
2. Sample observed windows from X_q.
3. Embed windows with the fixed ψ.
4. Score each enabled model by capability-region distance.
5. Fuse window/channel ranks.
6. Return one TSFM for downstream forecasting; candidate TSFMs are not forwarded.

INSERT($m_{\text{new}}, P_t^\ell, I_t^\ell$)
Reads: old capability map, fixed probe workload.
Writes: P_{t+1}^ℓ, I_{t+1}^ℓ.

1. Register metadata and compatibility constraints.
2. Run m_{new} only on D_p.
3. Append new error column to P_t^ℓ.
4. Recompute local-best assignments and affected entries.
5. Incorporate I_{t+1}^ℓ as the updated capability representation.

Figure 7: Operation contracts and accounting boundary. PROFILE materializes metric-specific capability maps and capability representations; ROUTE reads the current representation and returns one enabled TSFM without evaluating candidates; INSERT profiles only the incoming TSFM and incorporates the updated representation. ROUTE-stage cost excludes the final inference of the selected TSFM.

The model pool stores checkpoint identity, release order, compatibility metadata, and output capabilities. The capability map preserves the complete anchor-by-model loss matrix; the capability representation associates fixed anchor embeddings with metric-specific local best models. When new evidence is appended, capability assignments do not require re-embedding anchors or re-evaluating old models. Figure 7 specifies the read/write boundaries. ROUTE reads the current representation while the next update is prepared.

B.3 Cost Accounting

Table 5 separates comparison from the selected model's final inference. Here F_q, F_p, F_s denote task, probe-anchor, and sampled-window forward costs; $\bar{F}_q$ and $\bar{F}_s$ are averages. G_q is meta-feature extraction, E_ψ representation cost, N_{lab} the labeled-pool size, and C_{train} selector training cost. Δ_t denotes affected update work. The logarithmic retrieval expression is the capability-retrieval cost model; exact per-model distance retrieval can require scanning additional anchors and is not a worst-case guarantee independent of representation or zoo size. The empirical runtime plots measure the implemented path separately. The full comparison table is retained here so that the symbols used by the asymptotic comparison are defined next to their rows.

Table 5: Asymptotic comparison of online model selection and model-arrival adaptation. The selected model's final forecast is excluded. TSRouter shifts candidate evaluation from ROUTE to PROFILE/INSERT, so inference-time selection uses the fixed representation and model arrival requires only new-model profiling on anchors plus affected capability-region updates.

Strategy	Online candidate execution	ROUTE-stage latency	INSERT adaptation time	Main implication
Static / Recent	0	$O(1)$	$O(1)$ model-pool update	Cheap but not task-adaptive.
Full-Zoo Forward	M_t full forwards	$O(M_t\bar{F}_q)$	$O(1)$ model-pool update	Accurate upper bound; online cost grows with zoo size.
Task-Probe Forward	sM_t sample forwards	$O(sM_t\bar{F}_s)$	None	Moves selection cost to each task.
Relabel-Retrain selector	0 at ROUTE	$O(G_q + M_t)$	$O(N_{\text{lab}}\bar{F}_q(m_{\text{new}}) + C_{\text{train}})$	New model is forwarded on the labeled pool before relabel/retrain.
TSRouter	0	$O(sC_q(E_\psi + \log n))$	$O(nF_p(m_{\text{new}}) + \Delta_t)$	New model is profiled only on anchors; INSERT updates affected capability regions.
TSRouter-fast	0	TSRouter $+O(M_t)$	TSRouter	Adds runtime-aware reweighting from stored logs.

C DATA, EVALUATION PROTOCOL, AND LEAKAGE

TSFM-ZooBench augments GIFT-Eval with a release-ordered model pool and a common routing interface. Its 97 configurations are labeled forecasting tasks, not production task volume. Table 6 describes the domain and horizon mix.

The seven domains and ten frequencies create heterogeneous selection conditions. The horizon mix distinguishes repeated evaluation configurations from distinct source datasets: there are 23 datasets, and a dataset may contribute multiple frequency–horizon tasks. Quality metrics use the labeled configurations, while route/insert replay measures operations that do not require future labels. Additional replay volume therefore does not enlarge the statistical quality sample.

All models receive at most 512 context points. The main MASE model pool is configured so that every model is callable for each task; the compatibility mask is consequently all-true. Multivariate tasks are sampled channel-wise for routing, but the final forecast uses each model's native variate interface. Candidates lacking native probabilistic outputs remain valid for MASE. CRPS uses the benchmark's output-conversion interface and is auxiliary to the MASE-oriented ranking objective.

C.1 INFORMATION BOUNDARY

PROFILE anchors are sampled from public pretraining sources associated with Chronos, Moirai, and TimesFM, separately from evaluation tasks. Their labels support offline capability measurement. ROUTE receives observed task context and the capability representation; future targets and exhaustive test losses enter only after selection to compute metrics and oracle references. The fixed probe split, source-domain manifest, anchor provenance, and metric-specific partitions are necessary reproduction artifacts. This separates the router's information access but does not certify the pretraining contents of every public TSFM.

C.2 METRICS AND TIMING

Regret and Rank refer to MASE throughout; Task WR@k tests membership of the one selected model in the true Top-k set. Random uses five fixed seeds. Core-ROUTE excludes common sampling/preprocessing, while ROUTE includes it and E2E additionally includes selected-model inference. Timing records must distinguish measured, fallback-derived, predicted, zero-filled, and missing entries. Online summaries use P95; background insertion uses a mean. Experiments use an RTX 3090 with model-dependent dynamic batch sizes under the context/horizon constraints.

Table 6: Coverage of the labeled TSFM-ZooBench routing workload. The benchmark contains 97 labeled routing tasks over 23 GIFT-level datasets; each task is a dataset–frequency–horizon configuration. Horizon mix reports short/medium/long configurations, and Variates reports the channel-count range.

Domain	Tasks	Frequencies	Horizon mix (S/M/L)	Variates
Econ/Fin	6	A, Q, M, W, D, H	6/0/0	1
Energy	32	10T, 15T, D, H, W	16/8/8	1–7
Healthcare	5	D, M, W	5/0/0	1
Nature	15	10T, D, H, M, W	9/3/3	1–21
Sales	4	D, M, W	4/0/0	1
Transport	15	15T, 5T, D, H	7/4/4	1
Web/ CloudOps	20	10S, 5T, H	8/6/6	2–7
Total	97	10 frequencies	55/21/21	1–21

D Models, Baselines, and Configurations

Table 1 lists one representative per family and all size variants in the 20-model pool. The columns reveal distinct family strengths across MASE, rank, capability coverage, CRPS, and inference time; no single representative leads every criterion.

The representative model pool table is reported before the main results in Section 4; this appendix records model access and configuration details.

D.1 Baseline Access and Adaptation

Table 7 separates online candidate evaluation from shared profile evidence. Current-best is a retrospective test-label oracle and is excluded from deployable comparisons; Task-probe evaluates candidates on sampled observed windows. AutoForecast, AutoXPCR, and the SimpleTS-style reproduction require target updates or retraining as the model zoo changes, whereas TSRouter appends losses and updates fixed-anchor capability assignments. Its "No" arrival-action entry refers to learned-selector retraining, not zero update cost.

The default TSRouter uses 3,000 clustered anchors, fixed Fourier features, 20 windows, all-channel fusion, and the MASE local-best partition; TSRouter-fast uses the same evidence with $\lambda = 0.5$. The ablation table changes one component at a time and reports the resulting trade-offs.

E Additional Evidence for Capability Transfer

E.1 Model-Specific Capability Regions

The detailed model-specific capability-region heatmap and its interpretation are presented in Section 4.2; Table 8 retains the complete profile-holdout diagnostics in this appendix.

E.2 Zoo-Growth Adaptation Cost

Figure 6 reports the cost trace referenced in Section 4.3. ROUTE-stage P95 separates common task-window sampling from method-specific comparison, while INSERT reports mean offline work per arriving model. TSRouter pays anchor evaluation for the incoming model and updates affected capability regions; Task-probe additionally forwards candidates on sampled windows. Stages 5–20 are measured on the release stream, whereas 40 and 80 are projected stress points and do not establish quality at a larger zoo.

Table 7: Fairness and data-access contract for selector baselines. All methods select with observed task context only and cannot use future target labels at ROUTE time; all deployable methods ultimately return one Top-1 TSFM. "No" for TSRouter arrival action means no selector retraining, while fixed-anchor capability assignments are still updated.

Method	Online fwd.	Arrival action	Profile use	Output
Random/Recent	No	No	None	Static model
Profile-probe	No	Update	Shared profile	Static model
Task-probe	Yes	No	Route windows	Ranking
AutoForecast	No	Yes	Shared profile	Ranking
AutoXPCR	No	Yes	Profile+runtime	Ranking
SimpleTS	No	Yes	Shared profile	Ranking
TSRouter	No	No	Anchor capability representation	Ranking

Table 8: Profile-holdout capability diagnostics and task-level local best model recovery. Anchor WR@k measures whether the true anchor-level local best model appears in the Top-k models predicted from profile-holdout anchors. Capability-region purity measures how often anchors assigned to a model are actually best served by that model. Within-region rank is the mean true MASE rank of the region-associated model on its assigned anchors; lower is cleaner. Task WR@k reports downstream local best model recovery after ROUTE aggregation.

Anchor set	Encoder	Anchor WR@1 ↑	Anchor WR@3 ↑	Region purity ↑	Within-region rank ↓	Task WR@1 ↑	Task WR@3 ↑
Clustered	Fourier (main)	**0.446**	**0.620**	**0.4464**	**5.0594**	**0.268**	**0.567**
Random	Fourier	0.396	0.573	0.3960	5.4068	0.247	**0.567**
Clustered	Identity	0.076	0.284	0.1831	9.7625	0.206	0.443
Clustered	MLP	0.437	0.592	0.4366	5.7719	0.227	0.433
Clustered	SimpleTS-TS2Vec	0.393	0.561	0.3931	6.6872	0.186	0.392

E.3 Routing and Inference Cost Breakdown

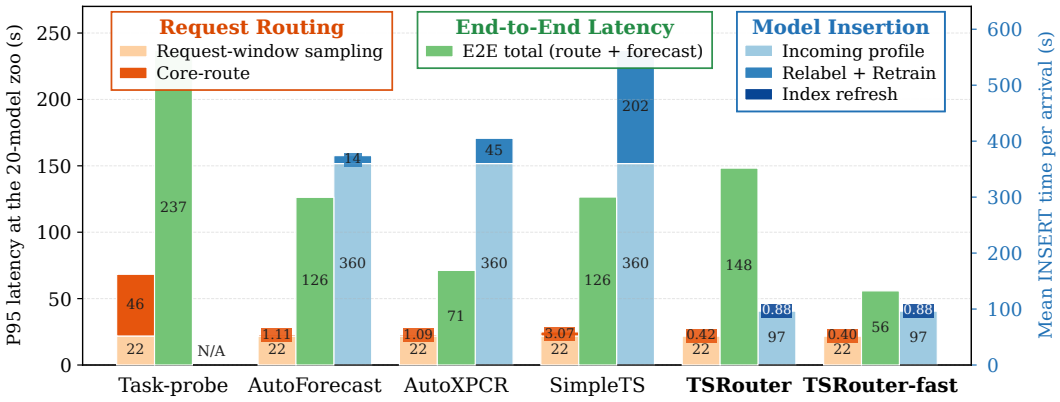


Figure 8: Breakdown of ROUTE-stage latency, E2E forecast latency, and INSERT adaptation time on the 20-model zoo. ROUTE P95 separates task-window sampling from method-specific Core-ROUTE comparison; E2E P95 further adds selected-model inference. INSERT is the mean per-arrival cost. TSRouter updates capability assignments without relabeling, selector retraining, or old-profile recomputation, whereas Task-probe includes sampled-window candidate evaluation inside Core-ROUTE.

Figure 8 explains why inexpensive routing need not produce the lowest E2E latency: the selected TSFM's inference can dominate. Task-probe includes candidate forwarding inside Core-ROUTE. TSRouter performs retrieval and rank aggregation there, then executes the chosen model. The runtime prior changes which model is chosen, providing the alternative operating point shown in the main

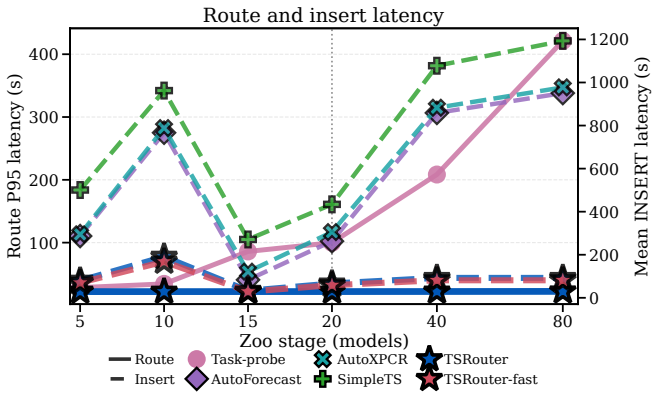


Figure 6: Zoo-growth adaptation cost. ROUTE P95 uses the left axis and mean INSERT time uses the right axis. Stages 5–20 are measured release-order stages; 40 and 80 are projected stress tests. The trace separates online candidate evaluation from bounded model-arrival adaptation and should not be read as a quality curve for an 80-model zoo.

text. Offline INSERT still evaluates the arriving TSFM on all fixed anchors; old profiles are reused rather than recomputed.

F Related Work

F.1 Zero-Shot Forecasting and Heterogeneous Model Strengths

Foundation models and forecasting benchmarks establish zero-shot reuse across diverse tasks (Qiu et al., 2024; 2025; Aksu et al., 2024), but a leaderboard alone does not specify which model to choose before an unseen task's future is available. TSRouter studies this selection layer with a changing candidate set.

F.2 Transferability Estimation and Model Selection

LEEP and LogME estimate pretrained-model suitability from task evidence (Nguyen et al., 2020; You et al., 2021); task embeddings and model-zoo methods organize related transfer information (Achille et al., 2019; Zhang et al., 2023; Li et al., 2024). Their settings differ from the present forecasting protocol. Our relevant distinction is the evidence interface: a task-time candidate evaluation, a learned task-to-model mapping, or retrieval from precomputed capability regions. The evaluated supervised and resource-aware selectors provide concrete comparisons under the shared TSFM protocol.

F.3 Representation-Based Retrieval

TS2Vec learns transferable time-series representations (Yue et al., 2022). A representation useful for prediction or classification is not necessarily optimal for model-capability retrieval; the encoder ablations test that distinction directly. TSRouter uses a fixed representation and preserves real probe windows as model-supporting examples. Its incremental property comes from appending measured model losses in this fixed space, rather than learning a new representation whenever the zoo changes.

G Limitations and Future Directions

The capability landscape is conditional on the probe corpus, representation, loss metric, and current candidate set; best-only assignments omit near-optimal alternatives, and models without local-best anchors lack finite distance support. Cross-benchmark transfer, unseen source domains, and fully held-out model families require separate experiments. The 40/80-model cost points are projections rather than enlarged-zoo quality results, and future releases may change both capability assignments and the accuracy–latency trade-off.