TimeRouter: Efficient and Adaptive Routing of Time-Series Foundation Models

Anonymous Authors[1]

Abstract

Time-series foundation models (TSFMs) are increasingly explored as predictive experts within emerging agentic time-series systems. However, TSFMs exhibit heterogeneous inductive biases, and no single model consistently dominates across forecasting regimes, making expert selection a critical challenge. Existing systems often delegate this decision to LLM-based controllers, incurring substantial inference overhead. We present TimeRouter, an efficient routing framework that leverages empirical complementarity across a pool of pretrained TSFMs through lightweight discriminative routing, selective gating, and ensemble fallback. Concretely, TimeRouter combines a learned routing head, a selective gate, and an ensemble fallback, enabling adaptive expert selection without invoking an LLM at inference time. TimeRouter achieves state-of-the-art performance on the GIFT-EVAL leaderboard, with an LB MASE of **0.6765**. Beyond benchmark performance, our ablation studies provide empirical insights into TSFM routing design, highlighting the importance of pool composition and selective gating. Taken together, these results position TimeRouter as a modular and lightweight routing layer for future agentic time-series systems built upon foundation-model pools.

1. Introduction

The last three years have seen rapid growth in time-series foundation models (TSFMs): Lag-Llama (Rasul et al., 2024), Chronos (Ansari et al., 2024; 2025), TimesFM (Das et al., 2024), TiRex (Auer et al., 2025), Moirai (Woo et al., 2024; Liu et al., 2024; 2025a), Sundial (Liu et al., 2025b), PatchTST-FM (Wen et al., 2026), FlowState (Graf et al., 2025), TTM (Ekambaram et al., 2024), and others. Each is pretrained on a distinct corpus and embodies distinct architectural choices; recent analysis by Yu et al. (2025) identifies three design axes (patch size, embedding type, training loss) along which these choices induce orthogonal inductive biases. As a result, no single design point is optimal across the full range of forecasting regimes: which TSFM performs best varies systematically with sampling frequency, forecast horizon, domain, and noise structure. The practical challenge is therefore not identifying a universally best TSFM, but adaptively selecting the right TSFM for each input, which naturally leads to a routing problem.

TSFMs as components in agentic systems. TSFMs are increasingly explored as components within emerging agentic time-series systems that select or combine multiple TSFMs at inference time. TimeCopilot (Garza & Rosillo, 2025) orchestrates feature analysis and model selection via a generic LLM agent; MoiraiAgent (Salesforce AI Research, 2025) uses a fine-tuned Qwen-2.5-3B for per-series expert selection; TSOrchestra (Cao et al., 2025) uses an R1-style fine-tuned LLM for ensemble orchestration over a multi-FM pool. Recent non-LLM approaches explore complementary directions: Synapse (Das et al., 2025) performs timestamp-level adaptive arbitration by dynamically reweighting the full TSFM pool, while ZooCast (Shi et al., 2025) performs task–model matching through embedding and similarity-based Top-K selection. These systems demonstrate the practical value of adaptive coordination across TSFMs, spanning LLM-based orchestration, adaptive ensemble arbitration, and embedding-based model matching. However, a lightweight discriminative routing layer for adaptive expert selection across TSFMs remains largely unexplored.

Theoretical basis. TimeRouter is motivated by two complementary ideas: stacked generalisation and selective prediction. *Stacked generalisation* (Wolpert, 1992) trains a second-level model over first-level outputs treated as features, motivating our use of per-FM cross-validation scores and downsampled forecasts as routing features, together with a CV-inverse-weighted ensemble fallback. *Selective prediction and learning-to-defer* (Geifman & El-Yaniv, 2017; Mozannar & Sontag, 2020; Verma & Nalisnick, 2022) establish that confidence-thresholded classifiers admit controllable risk–coverage tradeoffs. TimeRouter adopts this principle

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

through a selective gate that routes low-confidence inputs to the ensemble fallback rather than committing to a single expert.

TimeRouter. TimeRouter uses a one-vs-all classifier head to produce routing scores over a pool of TSFMs from a feature representation combining context information and base-model outputs. At inference, the classifier's decision-space margin and the pool's forecast-space diversity jointly drive a selective gate: low-confidence inputs are deferred to a CV-inverse-weighted ensemble fallback, while confident inputs commit to the classifier's argmax. On GIFT-EVAL (Aksu et al., 2024), TimeRouter achieves **LB MASE 0.6765**, a new state-of-the-art on the leaderboard.

Contributions. (i) We propose TimeRouter, an efficient discriminative routing framework for adaptive expert selection across time-series foundation models. (ii) On GIFT-EVAL, TimeRouter achieves LB MASE 0.6765, a new state-of-the-art on the leaderboard. The ablation studies further provide empirical insights into TSFM routing design. (iii) TimeRouter provides a modular and efficient routing layer for future agentic time-series systems built upon foundation-model pools.

2. Method

Problem definition. Given a fixed pool of K frozen TSFMs $\mathcal{F} = \{F_1, \dots, F_K\}$, each F_k maps a univariate context $x \in \mathbb{R}^T$ to a point forecast $F_k(x) \in \mathbb{R}^H$ over the next H steps. A deterministic ensemble combiner $\mathrm{Ens} : \mathbb{R}^{H\times K} \to \mathbb{R}^H$ is fixed in advance. A *router* is a policy $\pi(x; \mathcal{F}) \in \{1, \dots, K, \mathrm{Ens}\}$ producing the forecast

$$\hat{y}_\pi(x) = \begin{cases} F_{\pi(x)}(x) & \pi(x) \in [K], \\ \mathrm{Ens}\big(F_1(x), \dots, F_K(x)\big) & \pi(x) = \mathrm{Ens}. \end{cases} \quad (1)$$

We seek a router that minimises the expected per-row loss

$$\pi^\star = \arg\min_\pi \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\ell\big(\hat{y}_\pi(x), y\big)\big], \quad (2)$$

where $\mathcal{D}$ is the training distribution and ℓ is the per-row MASE.

Routing head and training objective. TimeRouter parameterises π by a one-vs-all (OvA) classifier over a fixed feature map $\phi(x) \in \mathbb{R}^d$ that concatenates context statistics (trend, seasonality, autocorrelation, length, ...), per-FM context-tail cross-validation scores, and downsampled per-FM forecasts; the latter two blocks are stacking-style features in the sense of Wolpert (1992). For each $k \in [K]$, a binary classifier $g_{\theta_k} : \mathbb{R}^d \to [0, 1]$ predicts whether F_k is the oracle-best FM on input x; denoting the oracle label

$$k^\star(x, y) = \arg\min_{k\in[K]} \ell\big(F_k(x), y\big), \quad (3)$$

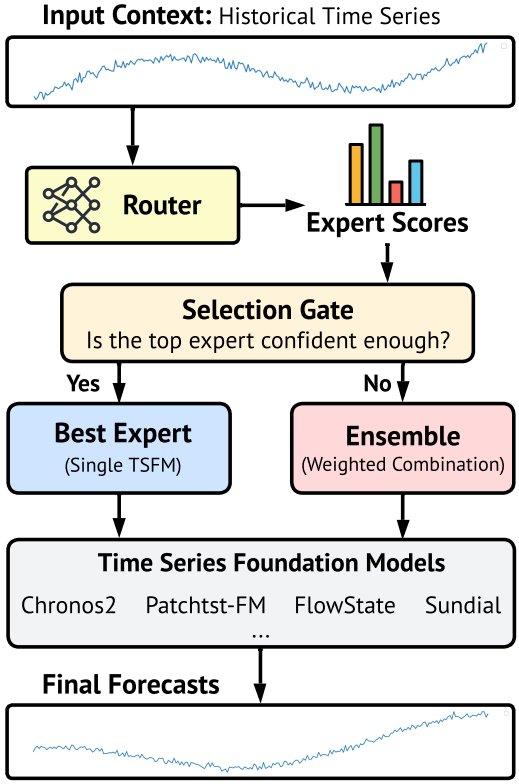

Figure 1. Overview of TimeRouter. Given an input context, the router produces routing scores over a pool of time-series foundation models (TSFMs). A selective gate determines whether to commit to the top expert or defer to an ensemble fallback when confidence is low, enabling adaptive expert selection across heterogeneous forecasting regimes.

each binary classifier is trained by minimising the expected binary cross-entropy

$$\theta_k^\star = \arg\min_{\theta_k} \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\mathrm{BCE}(g_{\theta_k}(\phi(x)), \mathbb{1}\{k^\star = k\})\big]. \quad (4)$$

At inference, the K classifier scores are L_1-normalised into a score vector $p(x) = (p_1(x), \dots, p_K(x))$. The router commits to $\arg\max_k p_k(x)$ when the gate trusts the prediction and defers to Ens otherwise.

Two scalar signals are computed at inference time from the routing scores and pool forecasts:

$$\text{margin: } m(x) = p_{(1)}(x) - p_{(2)}(x), \quad (5)$$

$$\text{diversity: } d(x) = H^{-1} \sum_{t=1}^{H} \mathrm{std}_k(F_k(x; t)/s(x)), \quad (6)$$

where $p_{(j)}$ is the j-th order statistic of the score vector $p(x)$, $F_k(x; t)$ is FM k's point forecast at horizon step t, and $s(x)$ is the per-series context scale. The margin lives in *decision space*; diversity lives in *forecast space* and is large on inputs where the FMs disagree about the future.

Low diversity indicates that the pool forecasts are already highly consistent, in which case committing to a single FM offers limited advantage over the ensemble fallback. Given thresholds $(\tau_m, \tau_d) \geq 0$, the gate routes

$$\pi(x) = \begin{cases} \text{Ens} & \text{if } m(x) < \tau_m \text{ or } d(x) < \tau_d, \\ \arg\max_k p_k(x) & \text{otherwise.} \end{cases} \tag{7}$$

Thresholds (τ_m, τ_d) are selected on training-split OOF (§3).

Ensemble combiner. The deployed combiner is a CV-inverse-weighted average:

$$\text{Ens}\big(F_1(x), \ldots, F_K(x)\big) = \sum_{k=1}^{K} w_k(x)\, F_k(x), \tag{8}$$

$$w_k(x) \propto \frac{1}{\text{CV\_score}_k(x) + \epsilon},$$

where CV_score_k is the context-tail single-window CV-MASE for FM k and weights are normalised to sum to one. Alternative combiners (unweighted mean, per-step median, inverse-CRPS) plug into the same gate without changing the head or threshold-tuning procedure.

3. Experiments

Benchmark and pool. We evaluate on GIFT-EVAL (Aksu et al., 2024), a 97-task forecasting benchmark with a public leaderboard. As the foundation-model pool, we use four checkpoints from multiple forecasting paradigms: Chronos-2 (Ansari et al., 2025), FlowState (Graf et al., 2025), PatchTST-FM (Wen et al., 2026), and Sundial (Liu et al., 2025b). These models are chosen for their strong standalone leaderboard performance and complementary forecasting behaviour. All pool members remain frozen during routing-head training.

Implementation details. The feature map has d=305 dimensions for our four-FM pool (165 pool-independent dimensions plus 35 per FM, of which 3 are CV statistics and 32 are forecast-snippet buckets; block-level breakdown in Appendix A). The one-vs-all classifier uses XGBoost (Chen & Guestrin, 2016) for each binary g_{θ_k}, one per FM. We fit S=5 seeds differing only in `random_state`; each seed's K classifier scores are L_1-normalised into a per-seed score vector and the S per-seed vectors are averaged at inference time. The gate thresholds (τ_m, τ_d) are selected by a grid sweep on a 5-fold task-grouped `GroupKFold` OOF split.

Main result. Table 1 compares TimeRouter against the strongest single-FM and routing baselines on the GIFT-EVAL leaderboard. TimeRouter achieves **LB MASE** 0.6765, a new state-of-the-art on the leaderboard; it improves over the strongest single FM (Chronos-2 at 0.6978) by ∼200 bp and edges the strongest LLM-judge router (TSOrchestra at 0.6768) by ∼3 bp while incurring no LLM inference cost at the foundation-model routing step.

Method	LB MASE
Foundation models (single)	
Chronos-2 (Ansari et al., 2025)	0.6978
TimesFM-2.5 (Das et al., 2024)	0.7050
PatchTST-FM (Wen et al., 2026)	0.7069
TiRex (Auer et al., 2025)	0.7158
FlowState (Graf et al., 2025)	0.7262
Moirai2 (Liu et al., 2025a)	0.7281
Sundial (Liu et al., 2025b)	0.7502
TTM-R2 (Ekambaram et al., 2024)	1.0196
Agentic systems	
TSOrchestra (Cao et al., 2025)	0.6768
MoiraiAgent (Salesforce AI Research, 2025)	0.6887
Credence (leaderboard entry)	0.6907
ZooCast (Shi et al., 2025)	0.6920
Synapse (Das et al., 2025)	0.6986
TimeCopilot (Garza & Rosillo, 2025)	0.7051
TimeRouter	**0.6765**

Table 1. GIFT-EVAL LB MASE comparison (lower is better). LB MASE is the geomean across 97 tasks of (method MASE / Seasonal-Naive MASE).

Efficiency. TimeRouter trains in ∼ 110 s for a 20-expert pool (20 one-vs-all XGBoost classifiers; ∼ 155K training rows) and incurs only 9.9 ms inference overhead per series. Its lightweight routing design enables rapid adaptation as the TSFM ecosystem evolves: newly released foundation models can be added to the pool and the routing head re-trained in minutes, without updating any TSFM or invoking an LLM-based orchestration loop. Compared with recent agentic-routing systems, TimeRouter substantially reduces routing overhead while preserving the flexibility of multi-expert coordination. Additional implementation and hardware details are provided in Appendix B.

Method	Additional learning cost	Routing latency / series
TSOrchestra	fine-tune 3B LLM	$\geq$ 472.6 ms
MoiraiAgent	fine-tune 3B LLM	472.6 ms
TimeCopilot	no training; LLM agent	$\geq$ 450 ms
Synapse	training-free	—
ZooCast	train embedding selector	—
TimeRouter	∼110 s	9.9 ms

Table 2. Efficiency comparison against published routing and agentic forecasting systems (TSOrchestra (Cao et al., 2025), MoiraiAgent (Salesforce AI Research, 2025), TimeCopilot (Garza & Rosillo, 2025), Synapse (Das et al., 2025), ZooCast (Shi et al., 2025)). Routing latency excludes foundation-model forecasting time for all methods.

4. Ablations

4.1. Selective gate ablation

Holding the one-vs-all classifier and the CV-inverse-weighted fallback fixed, we compare the deployed selective gate against a no-gate variant that always commits to the classifier argmax. Figure 2 reports per-term LB MASE on GIFT-EVAL.

The selective gate improves overall performance by $+13$ bp, but the effect is highly term-dependent: it improves long-horizon tasks by $+90$ bp and medium-horizon tasks by $+14$ bp, while slightly regressing on short-horizon tasks (-14 bp). This matches the intuition that forecast uncertainty grows with horizon, making cross-FM disagreement more informative on longer-horizon inputs and ensemble fallback more beneficial there. On short horizons, the classifier argmax is already strong and the gate provides limited additional benefit.

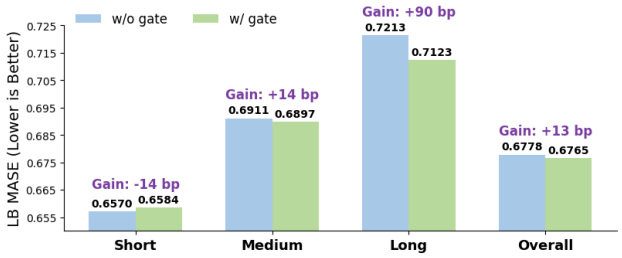

Figure 2. Gate ablation on TimeRouter, stratified by GIFT-EVAL term (LB MASE, lower is better). Same head and fallback in both variants.

Head	LB MASE	Δ (bp)
XGBoost(deployed)	0.6765	—
LightGBM	**0.6762**	-3
Random Forest	0.6776	$+11$
MLP	0.6787	$+22$
Logistic Regression	0.6836	$+71$

Table 3. Head ablation on TimeRouter. Same features, one-vs-all structure, and gate; only the binary classifier varies. Δ is in bp relative to XGBoost.

4.2. Routing head ablation

We compare five binary classifier families for the one-vs-all stage (Table 3), all sharing the same feature map, one-vs-all structure, and selective gate. The four non-linear families, XGBoost, LightGBM, Random Forest, and a 2-layer MLP, cluster within a narrow 25 bp band (0.6762–0.6787), while logistic regression, the only linear classifier, degrades substantially ($+71$ bp behind XGBoost). This suggests that routing performance depends primarily on modeling non-linear feature interactions rather than on the specific classifier family. We therefore use XGBoost as the default routing head.

4.3. Pool composition ablation

To assess the contribution of each pool member, we ablate the four-FM pool by evaluating all $\binom{4}{3}=4$ three-FM subsets and all $\binom{4}{2}=6$ two-FM subsets. Table 4 reports results for every subset, each using the same one-vs-all classifier, selective gate, and ensemble fallback as the full four-FM configuration.

Pool	LB MASE
Full four-FM pool	**0.6765**
$K=3$ subsets	
drop FlowState	0.6807
drop PatchTST-FM	0.6814
drop Sundial	0.6834
drop Chronos-2	0.6876
$K=2$ subsets	
Chronos-2 + Sundial	0.6847
Chronos-2 + PatchTST-FM	0.6863
Chronos-2 + FlowState	0.6888
FlowState + PatchTST-FM	0.6889
PatchTST-FM + Sundial	0.6938
FlowState + Sundial	0.7082

Table 4. Pool ablation over all three- and two-FM subsets.

Chronos-2 is the critical anchor. Dropping Chronos-2 yields the worst $K=3$ subset (0.6876, $+111$ bp behind the full pool); dropping any of the other three FMs costs only $+42$ to $+69$ bp. At $K=2$ the asymmetry persists: all three subsets that retain Chronos-2 score 0.6847–0.6888, while every subset without Chronos-2 scores ≥ 0.6889, with FlowState+Sundial the worst at 0.7082. Pool size matters too, best-to-best LB MASE degrades from 0.6765 ($K=4$) to 0.6807 (best $K=3$) to 0.6847 (best $K=2$), each step costing $\sim$40 bp.

5. Conclusion

We presented TimeRouter, an efficient discriminative routing framework for adaptive expert selection across time-series foundation models, achieving state-of-the-art LB MASE on the GIFT-EVAL leaderboard without invoking an LLM at routing time. Our ablation studies provide several insights for future TSFM routing systems: pool composition and pool size both matter; the selective gate is most beneficial on long-horizon tasks where forecast disagreement becomes informative; and classifier family matters little as long as non-linear feature interactions can be modeled effectively. Because the routing layer is lightweight and easy to adapt, newly released TSFMs can be integrated into the pool without retraining existing foundation models, positioning TimeRouter as a modular routing layer for future agentic time-series systems.

References

Aksu, T., Woo, G., Liu, J., Liu, X., Liu, C., Savarese, S., Xiong, C., and Sahoo, D. Gift-eval: A benchmark for general time series forecasting model evaluation. *arXiv preprint arXiv:2410.10393*, 2024.

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Pineda Arango, S., Kapoor, S., et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Ansari, A. F. et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

Auer, A., Podest, P., Klotz, D., Böck, S., Klambauer, G., and Hochreiter, S. Tirex: Zero-shot forecasting across long and short horizons with enhanced in-context learning. *arXiv preprint arXiv:2505.23719*, 2025.

Cao, D., Gee, M., Liu, J., Wang, H., Yang, W., Wang, R., and Liu, Y. Conversational time series foundation models: Towards explainable and effective forecasting. *arXiv preprint arXiv:2512.16022*, 2025.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Das, A., Kong, W., Sen, R., and Zhou, Y. A decoder-only foundation model for time-series forecasting. *ICML*, 2024.

Das, S. S. S., Goyal, P., Parmar, M., Song, Y., Le, L. T., Miculicich, L., Yoon, J., Zhang, R., Palangi, H., and Pfister, T. Synapse: Adaptive arbitration of complementary expertise in time series foundational models. *arXiv preprint arXiv:2511.05460*, 2025.

Ekambaram, V., Jati, A., Nguyen, N., Sinthong, P., and Kalagnanam, J. Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. *arXiv preprint arXiv:2401.03955*, 2024.

Garza, A. and Rosillo, R. Timecopilot. *arXiv preprint arXiv:2509.00616*, 2025.

Geifman, Y. and El-Yaniv, R. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Graf, L. et al. Flowstate: A sampling-rate-invariant ssm-based time-series foundation model. *arXiv preprint arXiv:2508.05287*, 2025.

Liu, C. et al. Moirai 2.0: When less is more for time series forecasting. *arXiv preprint arXiv:2511.11698*, 2025a.

Liu, X. et al. Moirai-moe: Empowering time series foundation models with sparse mixture of experts. In *International Conference on Machine Learning (ICML)*, 2024. arXiv:2410.10469.

Liu, Y. et al. Sundial: A native flexible decoder transformer for time series. In *International Conference on Machine Learning (ICML, Oral)*, 2025b. arXiv:2502.00816.

Mozannar, H. and Sontag, D. Consistent estimators for learning to defer to an expert. In *International Conference on Machine Learning (ICML)*, 2020.

Rasul, K., Ashok, A., Williams, A. R., Ghonia, H., Bhagwatkar, R., Khorasani, A., Bayazi, M. J. D., Adamopoulos, G., Riachi, R., Hassen, N., Biloš, M., Garg, S., Schneider, A., Chapados, N., Drouin, A., Zantedeschi, V., Nevmyvaka, Y., and Rish, I. Lag-llama: Towards foundation models for probabilistic time series forecasting, 2024. URL https://arxiv.org/abs/2310.08278.

Salesforce AI Research. Moiraiagent: An agentic framework for context-aware time-series forecasting. Salesforce AI Research blog post, https://www.salesforce.com/blog/moiraiagent/, 2025.

Shi, H.-N., Huang, T.-J., Han, L., Zhan, D.-C., and Ye, H.-J. One-embedding-fits-all: Efficient zero-shot time series forecasting by a model zoo. *arXiv preprint arXiv:2509.04208*, 2025.

Verma, R. and Nalisnick, E. Calibrated learning to defer with one-vs-all classifiers. In *International Conference on Machine Learning (ICML)*, 2022.

Wen, Y., Gifford, W. M., Reddy, C., Nguyen, L. M., Kalagnanam, J., and Julius, A. A. Revisiting the generic transformer: Deconstructing a strong baseline for time series foundation models. *arXiv preprint arXiv:2602.06909*, 2026.

Wolpert, D. H. Stacked generalization. *Neural Networks*, 5 (2):241–259, 1992.

Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. Unified training of universal time series forecasting transformers. *ICML*, 2024.

Yu, A., Maddix, D. C., Ansari, A. F., Mahoney, M. W., et al. Understanding the implicit biases of design choices for time series foundation models. *arXiv preprint arXiv:2510.19236*, 2025.

A. Implementation Details

XGBoost head. `binary:logistic` with `max_depth=6`, `n_estimators=200`, `learning_rate=0.05`, `subsample=0.8`, `tree_method=hist`; one binary classifier per FM. We fit $S{=}5$ seeds differing only in `random_state`; each seed produces K class scores that are L_1-normalised into a per-seed score vector, and the S per-seed vectors are averaged at inference time.

Training data and filter. Training rows are drawn from the GIFT-EVAL training split (disjoint from the 97-task test split). Per-FM forecasts, oracle-best labels, and the feature vector $\phi(x)$ are precomputed once and shipped as parquet files, so the router sees only $(\phi(x), k^\star(x, y))$ at training time. We drop rows with $\mathrm{best\_mase} > 1.0$ (poor router headroom), $\mathrm{scale} < 0.01$ (degenerate near-constant series), or fewer than 2 pool members producing a valid forecast; this leaves $\sim$155K training rows over $\sim$93 tasks for the four-FM pool.

Gate threshold sweep. We sweep $\tau_m \in \{0.00, 0.02, 0.05, 0.07, 0.09, 0.10, 0.12, 0.15, 0.20, 0.25\}$ and $\tau_d \in \{0.0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.07, 0.10\}$, evaluating each (τ_m, τ_d) on a 5-fold task-grouped `GroupKFold` OOF split over the training set only. The selected thresholds for the four-FM pool are $(\tau_m, \tau_d) = (0.12, 0.02)$ and are fixed for all test-time evaluations.

Ensemble combiner Ens**.** CV-inverse-weighted average $\mathrm{Ens}(F_1, \ldots, F_K) = \sum_k w_k F_k$ with $w_k \propto 1/(\mathrm{CV\_score}_k + 10^{-8})$ and $\sum_k w_k = 1$. The per-FM CV scores are computed from context-tail validation windows for both training and test inputs and used as routing features as well as ensemble weights.

Feature map ϕ. The feature vector for each (series, cutoff) row concatenates four blocks; total dimension $d = 165 + 35K$ (d=305 for the four-FM pool). (i) *Context-window statistics* (31 dims): 18 time-series descriptors (mean, std, range, IQR, skewness, kurtosis, acf at lags 1, 5, 10, trend slope, diff statistics, zero-crossings, turning points, log length), 5 static metadata (horizon, series length, horizon/length ratio, frequency descriptor, number of available pool members), and 8 regime-shift descriptors. (ii) *Normalised context snippet* (128 dims): the input context normalised by its own mean/std and linearly resampled to 128 buckets. (iii) *Per-FM CV statistics* ($3K + 6$ dims): for each pool member, a single-window context-tail CV-MASE score, its rank within the pool, and its gap to the pool best ($3K$ dims), plus 6 pool-level CV aggregates. (iv) *Per-FM forecast snippets* ($32K$ dims): each pool member's forecast linearly resampled to $\mathrm{PRED\_LEN} = 32$ buckets and normalised by the context mean/std. Blocks (iii) and (iv) are stacking-style features in the sense of Wolpert (1992).

B. Efficiency Comparison

Table 5 compares additional learning cost, router scale, inference latency, and LB MASE against published routing and agentic forecasting systems. Compared with LLM-based orchestration approaches, TimeRouter replaces heavyweight reasoning agents with a lightweight discriminative router that trains in minutes on CPU hardware and serves at millisecond-level latency, while achieving state-of-the-art LB MASE among publicly reproducible methods on GIFT-EVAL.

Method	Additional learning cost	Router size	Infer.	LB MASE
TSOrchestra (Cao et al., 2025)	fine-tune reasoning LLM (R1-style)	3B LLM	$\geq$ 472.6 ms	0.6768
MoiraiAgent (Salesforce AI Research, 2025)	fine-tune 3B LLM	3B LLM	472.6 ms	0.6887
TimeCopilot (Garza & Rosillo, 2025)	no training; LLM agent	LLM-scale	$\geq$ 450 ms	0.7051
Synapse (Das et al., 2025)	training-free adaptive ensemble	—	—	0.6986
ZooCast (Shi et al., 2025)	train co-embedding selector	—	—	0.6920
TimeRouter (ours)	$\sim$ 110 s CPU	1.72 MB	**9.9 ms**	**0.6765**

Table 5. Additional learning cost, router size, inference latency, and LB MASE against routing-based and agentic forecasting systems. Dashes indicate quantities that are either not reported or not directly applicable.

Additional learning cost descriptors are sourced from each baseline's original paper or blog and are not directly comparable across methods. Inference costs are measured from publicly available implementations when reproducible code is available;

otherwise, they are estimated from the reported system configuration. Our measurements are conducted on a single Intel Xeon Gold 5318Y CPU with 16 threads per classifier using XGBoost 3.1.3.

B.1. Routing-overhead Benchmark Details

Table 5 compares routing/orchestration overhead only and excludes foundation-model forecasting time for all methods. Since different systems invoke different TSFM pools and forecasting backbones, including foundation-model forward passes would confound the comparison with unrelated implementation differences. We therefore isolate the latency introduced by the routing/orchestration layer itself.

All methods are evaluated on the same five GIFT-EVAL tasks: `ett1/W`, `saugeenday/M`, `us_births/M`, `M_DENSE/D`, and `hierarchical_sales/W`. Following the standard rolling-window evaluation protocol, these tasks contain a total of 585 forecasting rows. We report the rows-weighted average routing latency across all five tasks.

TimeRouter (ours). We measure routing overhead only, including feature extraction and router inference, excluding all foundation-model forward passes. Measurements are conducted using a CPU-based XGBoost router (XGBoost 3.1.3) on a single Intel Xeon Gold 5318Y CPU with 16 threads per classifier.

MoiraiAgent. MoiraiAgent publicly releases its inference pipeline, allowing direct reproduction. We reproduce the routing overhead on a single NVIDIA RTX A6000 GPU under the same evaluation setup. The reported latency isolates the LLM-agent orchestration component and excludes foundation-model forecasting time.

TSOrchestra. TSOrchestra does not release reproducible inference code for its LLM-routing version. The paper reports using the same Qwen2.5-3B reasoning model as MoiraiAgent. Since TSOrchestra additionally involves multi-round reasoning and orchestration, we conservatively estimate its routing overhead to exceed that of MoiraiAgent.

TimeCopilot. TimeCopilot does not release the exact GIFT-EVAL inference configuration used in the paper. The system reports using GPT-4o-mini as the orchestration LLM. Public benchmarks report a time-to-first-token latency of roughly 0.45–0.5 seconds for GPT-4o-mini, excluding additional tool-execution overhead. We therefore conservatively estimate the routing overhead to exceed 450 ms per forecasting row.

Synapse and ZooCast. Neither Synapse nor ZooCast publicly release reproducible inference code, and the implementation details provided in the papers are insufficient to reliably isolate routing overhead from forecasting execution. We therefore mark routing latency as unavailable for both methods.