

# Learning Agentic Time Series Forecasting via Trajectory Distillation and Memory-Based Reinforcement Learning

Xiaoyu Tao<sup>1</sup>, Mingyue Cheng<sup>1</sup>, Yaguo Liu<sup>1</sup>, Xingpeng Gao<sup>1</sup>,  
Qingsong Wen<sup>2</sup>, Qi Liu<sup>1</sup>, Shijin Wang<sup>3</sup>, Enhong Chen<sup>1</sup>

<sup>1</sup>State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China

<sup>2</sup>Squirrel Ai Learning <sup>3</sup>iFLYTEK Research, Hefei, China

{txytiny, ygliu, gaopingpeng}@mail.ustc.edu.cn

{mycheng, qiliuq1, cheneh}@ustc.edu.cn

qingsongedu@gmail.com, sjwang3@iflytek.com

## Abstract

Recent agentic time series forecasting (TSF) approaches use large language models (LLMs) to autonomously orchestrate analytical and forecasting tools and reason about temporal patterns using the resulting evidence. However, many such approaches rely on powerful LLMs in training-free, prompt-driven workflows, leaving open whether smaller models can learn these interaction and reasoning capabilities. In this paper, we introduce Cast-R1, an 8B-parameter agentic forecasting model trained through trajectory distillation and multi-turn reinforcement learning with structured memory. First, we develop a forecasting agent harness with specialized analysis and prediction tools to generate multi-turn interaction trajectories using a teacher LLM, which are subsequently evaluated and selected for supervised fine-tuning. Second, we apply group relative policy optimization (GRPO) across multi-turn forecasting episodes, with structured memory preserving diagnostic evidence and intermediate forecasts for state construction. Rewards based on prediction accuracy and temporal structure guide tool selection, model routing, and forecast refinement. A difficulty-aware curriculum combines sample filtering and progressive sample ordering during training. Extensive experiments on multiple datasets demonstrate the effectiveness of Cast-R1. These findings highlight the potential of forecasting-specific post-training to equip smaller LLMs with multi-turn, tool-augmented forecasting capabilities within structured workflows. <sup>1</sup>

## 1 Introduction

Agentic time series forecasting (TSF) uses large language models (LLMs) to orchestrate analytical and forecasting tools and reason about temporal patterns from the resulting evidence (Zhang et al.,

2025; Zhao et al., 2025). Through these interactions, an agent can characterize temporal patterns, select suitable forecasters, and refine predictions using accumulated evidence.

A prominent line of agentic forecasting research relies on powerful LLMs whose pretrained capabilities are elicited through prompting, without forecasting-specific policy training (Zhang et al., 2025; Zhao et al., 2025). Post-training has also advanced numerical forecasting and temporal reasoning (Niu et al., 2025; Zhou et al., 2026), as well as tool-augmented time series question answering (Wu et al., 2026). Building on these studies, we learn a single policy that jointly coordinates diagnostic tool selection, model routing, and numerical forecast revision using episode-local evidence.

We therefore ask: **How can forecasting-specific post-training enable a smaller LLM to coordinate tools and refine predictions using accumulated evidence?** The first challenge, learning from demonstrations, is that forecasting benchmarks provide numerical targets rather than demonstrations of diagnostic tool use, model routing, and forecast revision. This motivates generating, evaluating, and selecting teacher trajectories to supervise these decisions (Chen et al., 2023; Cui et al., 2026). The second challenge, learning from outcomes, arises because tool interactions change the evidence available to subsequent decisions, while forecasting quality is assessed only after the final prediction. Learning from these outcomes requires preserving decision-relevant evidence across turns (Hu et al., 2025) and using terminal forecasting feedback to improve the preceding decisions (Wang et al., 2025; Cheng et al., 2026).

We propose **Cast-R1**, an 8B-parameter agentic TSF model with a two-stage post-training framework. First, we develop a forecasting agent harness with specialized time series analysis and prediction tools, enabling a teacher LLM to generate multi-

<sup>1</sup>Our code is available at <https://github.com/Xiaoyu-Tao/Cast-R1-TS>.

turn trajectories. We evaluate and select these trajectories for trajectory distillation through supervised fine-tuning (SFT), initializing the student’s forecasting interaction policy. The state–response demonstrations teach the student to interpret tool feedback and follow the workflow.

Second, we combine structured memory-based state construction with multi-turn GRPO to optimize the initialized policy. The environment retains diagnostic summaries and intermediate model forecasts within each episode and incorporates them into subsequent decision states. Within the preparation–forecasting–refinement protocol, the learned decisions are tool selection, model routing, and numerical forecast revision. Terminal rewards for accuracy and temporal structure guide trajectory-level policy optimization across turns. A difficulty-aware curriculum supports training. Experimental results across diverse real-world datasets establish the effectiveness of Cast-R1, while ablation studies confirm the contributions of both SFT and RL to its performance. Together, these findings demonstrate that forecasting-specific post-training enables smaller LLMs to learn effective multi-turn decision-making and tool-use strategies for structured forecasting tasks.

Our main contributions are:

- We introduce Cast-R1, an 8B-parameter agentic TSF model that learns evidence-guided, multi-turn tool-use policies through post-training within a structured workflow.
- We develop an agent harness to generate, evaluate, and select teacher trajectories for supervised distillation of forecasting policies.
- We combine forecasting-specific memory-based state construction with multi-turn GRPO to optimize tool selection, model routing, and forecast refinement using diagnostic evidence, intermediate forecasts, and accuracy and temporal-structure rewards.

## 2 Related Work

### 2.1 LLM-Based Time Series Forecasting

Surveys categorize LLM forecasting by architecture, training, and evaluation (Xu et al., 2026), and time-series agents by tasks, tools, and memory (Chen et al., 2026). Pretrained models learn temporal patterns at scale, providing reusable representations across domains and prediction horizons.

Time-MoE (Shi et al., 2025) uses a mixture-of-experts architecture, while Chronos (Ansari et al., 2025) supports diverse forecasting settings. LLM-enhanced methods incorporate text and context (Xue and Salim, 2024; Gruver et al., 2023; Liu et al., 2024a). Time-LLM (Jin et al., 2024) reprograms time-series inputs for LLMs; TimeCMA (Liu et al., 2025) aligns temporal and textual modalities; TimeRAG (Yang et al., 2025b) retrieves historical sequences. Agentic frameworks introduce multi-step reasoning and tool interaction to coordinate forecasting decisions under evolving evidence (Jalori et al., 2025; Kim et al., 2026). TimeSeriesScientist (Zhao et al., 2025) coordinates diagnosis, model selection, validation, and reporting within a unified forecasting workflow. These methods motivate adaptive forecasting through state-aware reasoning and tool interaction (Shen et al., 2026). Beyond numerical accuracy, TemporalBench (Weng et al., 2026) evaluates historical understanding, contextual reasoning, and event-conditioned prediction.

### 2.2 Learning Tool-Using Agents

ReAct (Yao et al., 2023) interleaves reasoning, actions, and observations; FireAct (Chen et al., 2023) fine-tunes agents on teacher-generated trajectories. RAGEN (Wang et al., 2025) studies multi-turn RL for sequential decision-making in interactive environments; Agent-R1 (Cheng et al., 2026) supports step-level trajectories and context management across extended agent interactions. Augur (Cui et al., 2026) distills teacher-generated causal explanations for discrete trend prediction. Time-R1 (Zhou et al., 2026) combines supervised warm-up with RL for forecast-oriented reasoning; LangTime (Niu et al., 2025) uses PPO for autoregressive forecasting. TimeART (Wu et al., 2026) learns tool use from expert demonstrations for time-series question answering. ZARA (Li et al., 2026) grounds training-free motion time-series reasoning in knowledge and retrieval for activity recognition. HiAgent (Hu et al., 2025) maintains working memory via subgoal and observation summaries. MemCast (Tao et al., 2026) conditions frozen LLMs on hierarchical training-set forecasting memories. Cast-R1 combines trajectory distillation with multi-turn GRPO to learn diagnostic tool selection, model routing, and numerical revision. Its forecasting environment organizes episode-local diagnostics and intermediate forecasts as structured memory, while accuracy and temporal-structure rewards guide pol-

icy optimization within a preparation–forecasting–refinement protocol.

### 3 The Proposed Cast-R1

Cast-R1 learns forecasting policies through trajectory distillation and multi-turn GRPO with structured memory. Figure 1(a) presents its memory-guided forecasting workflow; Figure 1(b) summarizes two-stage policy learning.

#### 3.1 Sequential Forecasting Formulation

For a look-back length  $L$  and forecasting horizon  $H$ , let  $z = (\mathbf{x}_{t-L+1:t}, \mathbf{c}_{t-L+1:t}, u_t)$  denote the inputs and  $y = \mathbf{x}_{t+1:t+H}$  the target. Here,  $u_t$  denotes optional task context. For electricity price forecasting, it contains exogenous forecasts or known variables for the prediction window and textual information available at forecast origin  $t$ ; for the long-term tasks,  $u_t = \emptyset$ . Future target values are excluded from  $z$ .

We formulate forecasting as a sequence of decision steps  $k = 1, \dots, K$ , subject to a budget  $K \leq K_{\max}$ . At step  $k$ , the agent conditions on the policy-visible state  $s_k$  and generates a response  $a_k$  with tool calls or the final forecast. Each decision step corresponds to one agent response and may include multiple independent tool calls. Tool calls acquire diagnostic evidence or invoke forecasting models; the final response refines intermediate predictions and outputs the forecast. We denote the trajectory by  $\tau = ((s_1, a_1), \dots, (s_K, a_K))$  and its final prediction by  $\hat{y}(\tau)$ . An episode ends upon final forecast output or budget exhaustion. Training uses terminal rewards to optimize these decisions, while inference relies on the supplied inputs and tool-derived evidence.

#### 3.2 Memory-Based State Construction

Episode-local memory  $M_k = (D_k, Q_k)$  retains diagnostic outputs in Analysis History  $D_k$  and intermediate forecasts in Prediction Results  $Q_k$ , starting from  $M_1 = (\emptyset, \emptyset)$ . The prompt  $s_k = P(z, M_k)$  combines task inputs and context  $z$  with these two memory components.

$P$  selects stage-specific instructions according to the available diagnostics and forecasts. Early prompts include full task inputs; refinement prompts combine recent observations, task context, and memory. Prior agent responses and raw

tool messages are not additionally appended as conversation history. For nonterminal decisions:

$$\begin{aligned} a_k &\sim \pi_\theta(\cdot \mid s_k), \quad o_k = T(s_k, a_k), \\ M_{k+1} &= U(M_k, a_k, o_k), \\ s_{k+1} &= P(z, M_{k+1}), \end{aligned} \quad (1)$$

where  $T$  executes tool calls and returns  $o_k$ . The environment implements  $U$  to update memory and  $P$  to construct the next prompt from task inputs and retained evidence. Appendix C.2 details stage-aware prompt construction.

#### 3.3 Toolkit Design and Implementation

The toolkit supports evidence acquisition and initial forecast generation through two types of callable interfaces.

**Diagnostic Tools.** Five categories cover data quality assessment, global statistical characterization, structural and dynamic analysis, event-level summarization, and model adequacy and residual diagnostics. Their outputs summarize data issues, distributional and frequency-domain properties, regime changes, temporal events, and baseline residuals to support model selection and forecast refinement. Representative statistics and their definitions appear in Appendix C.1.

**Forecasting Model Invocation.** A unified interface exposes ARIMA, Chronos, PatchTST, and iTransformer as the candidate pool  $B$ :  $\tilde{y} = f_m(z; H)$ ,  $m \in B$ . The policy selects  $m$  using accumulated evidence, and the tool returns an  $H$ -step forecast for subsequent refinement. The pool combines statistical, pretrained, temporal, and cross-variable forecasting capabilities.

#### 3.4 Memory-Guided Agentic Forecasting

Figure 1(a) shows three prescribed stages between task input and final output: evidence acquisition (preparation), initial prediction (forecasting), and reflection and refinement. The policy selects tools, routes models, and revises forecasts using  $z$  and tool evidence, excluding future targets.

During preparation, diagnostic tools characterize the series and populate Analysis History; independent calls may execute in parallel. During forecasting, the agent selects a model using this evidence and stores its initial forecast in Prediction Results. During refinement, the same policy checks consistency and integrates recent observations, task

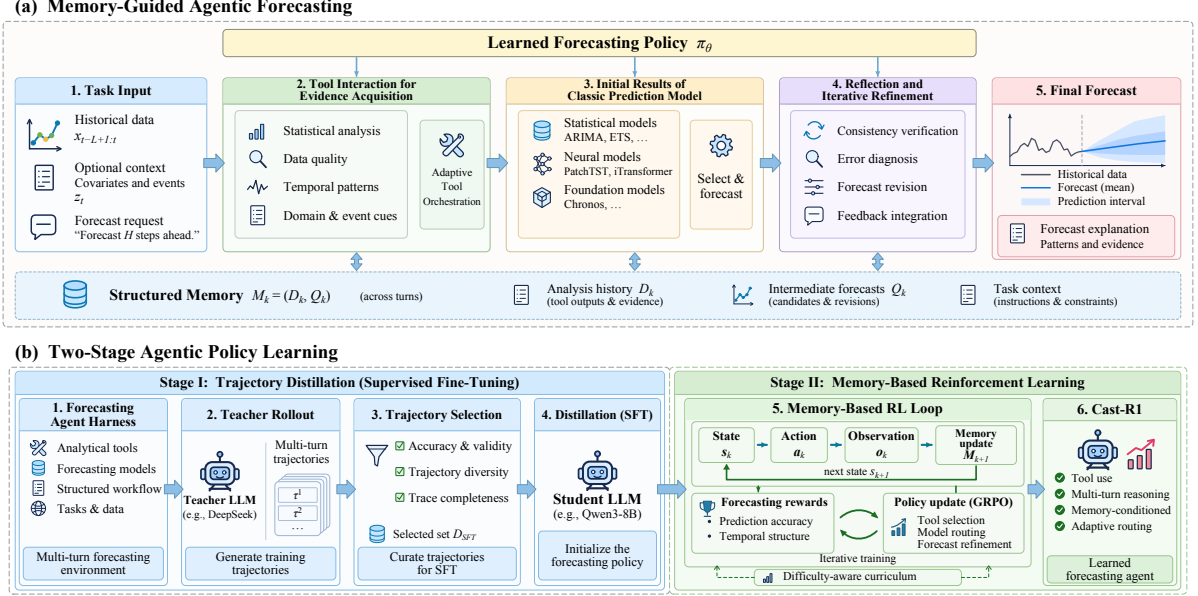


Figure 1: Overview of Cast-R1 as a learned agentic forecasting framework that combines structured memory for state construction with tool-augmented sequential decision-making and policy training.

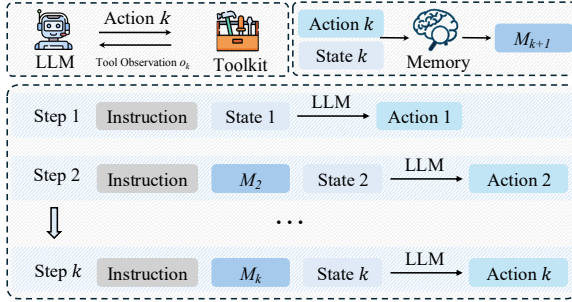


Figure 2: Structured memory for state construction. The environment updates  $M_k = (D_k, Q_k)$  with tool observations and constructs the next policy state.

context, and both memory components to revise the forecast and generate the final output:

$$\hat{y} = F_\theta(s_{\text{ref}}), \quad (2)$$

where  $s_{\text{ref}}$  is the refinement state and  $F_\theta$  denotes final forecast generation by the same policy  $\pi_\theta$ . Stage-aware prompts appear in Appendix C.3.

### 3.5 Two-Stage Agentic Policy Learning

Figure 1(b) shows two stages: teacher rollout and trajectory selection supply supervised fine-tuning (SFT); multi-turn GRPO then optimizes the initialized memory-conditioned policy using final-forecast feedback.

#### 3.5.1 Harness-Guided Trajectory Distillation

DeepSeek-V3.2 (DeepSeek-AI, 2025) generates multi-turn demonstrations of diagnostic tool use,

model selection, and forecast refinement in the forecasting harness, using only training-split windows from the ten datasets. Retained trajectories train the Qwen3-8B student through state-response supervision conditioned on accumulated tool feedback.

**Trajectory Evaluation and Selection.** We apply execution-validity filtering followed by forecasting-quality selection. Incomplete trajectories and those with failed tools, malformed outputs, or non-finite predictions are discarded. Valid trajectories are ranked by ascending final-forecast MSE within each dataset using training targets. We retain the 120 lowest-error trajectories per dataset, yielding 1,200 complete demonstrations in  $D_{\text{SFT}}$  with balanced dataset representation. Appendix D.2 details the completeness checks, allocation for dataset diversity, and SFT settings. Trajectory distillation minimizes the following SFT objective over demonstration trajectories  $\tau^* \sim D_{\text{SFT}}$ :

$$L_{\text{SFT}}(\theta) = -\mathbb{E}_{\tau^*} \left[ \sum_k \log \pi_\theta(a_k^* | s_k^*) \right]. \quad (3)$$

Here,  $s_k^*$  is the demonstrated state, including tool feedback, and  $a_k^*$  is the teacher response containing tool calls or the final forecast. Only teacher responses are supervised.

#### 3.5.2 Multi-turn RL with Structured Memory

We apply group relative policy optimization (GRPO) (Shao et al., 2024) to the distilled policy in

a multi-turn forecasting environment (Wang et al., 2025). The environment updates memory and constructs states as in Equation 1, while GRPO updates the shared policy  $\pi_\theta$ . Earlier tool choices therefore shape the evidence available to subsequent decisions. Relative advantages are estimated from terminal rewards of trajectories sampled for the same input, without a separate critic. The objective is to maximize expected terminal reward:

$$J(\theta) = \mathbb{E}[R(\tau, y)], \quad (4)$$

where the expectation is over training input–target pairs and policy rollouts, and  $R(\tau, y)$  evaluates the final forecast. For each input, GRPO samples  $G$  complete trajectories  $\{\tau_i\}_{i=1}^G$  using the rollout policy  $\pi_{\text{old}}$ . With  $R_i = R(\tau_i, y)$ , the group-relative advantage is:

$$\hat{A}_i = \frac{R_i - \bar{R}}{\sigma_R + \delta}, \quad (5)$$

where  $\bar{R}$  and  $\sigma_R$  are the group mean and standard deviation, and  $\delta > 0$  is a numerical stabilizer. The same trajectory-level advantage supervises agent-generated tokens across its turns, while tool outputs serve as environment observations. GRPO uses these advantages in clipped policy updates with KL regularization against a reference policy. Appendix D.3 gives the token-level formulation; Table 7 lists the training settings.

**Multi-view Reward Design.** The terminal reward combines numerical accuracy, temporal structure, and output completeness to guide forecasting decisions. For outputs with valid `<answer>` and `</answer>` tags, the reward is:

$$R_{\text{valid}} = p \left( w_{\text{len}} + \sum_j w_j u_j \right), \quad (6)$$

where  $p = \min(1, n_{\text{pred}}/n_{\text{gt}})$  uses the prediction and target lengths  $n_{\text{pred}}$  and  $n_{\text{gt}}$ . The sum ranges over  $j \in \{\text{pred}, \text{seas}, \text{trend}, \text{max}, \text{min}\}$ , covering prediction accuracy, seasonality, trend, and maximum/minimum alignment. Prediction, seasonal, and trend MSEs are mapped to scores in  $[0, 1]$  by  $q(e) = [1 + \log(1 + e)]^{-1}$  for  $e \geq 0$ ; alignment terms use extremum hit rates. Missing answer tags yield  $R = -1$ . Weights and scoring details appear in Appendix D.5. Rewards are computed exclusively on training-split samples using their targets; validation and test targets are excluded.

**Curriculum Learning.** The difficulty-aware curriculum in Figure 1(b) organizes training samples along two dimensions: prediction MSE of a reference forecaster and data-intrinsic complexity measured by normalized permutation entropy (Bandt and Pompe, 2002). The curriculum filters noise-dominated samples rather than selecting examples solely by high prediction error, and progressively introduces retained samples from simple trend and seasonal patterns to more complex temporal structures. Appendix D.4 defines the entropy statistic and distinguishes the reference forecaster from the trajectory teacher.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets.** We evaluate four ETT datasets (Zhou et al., 2021) and Wind (Qiu et al., 2024) for LTSF, and five regional EPF datasets (NP, PJM, BE, FR, and DE) (Lago et al., 2021). Table 2 reports their sizes, variables, and temporal resolutions. LTSF uses historical targets and covariates; EPF additionally provides forecast-horizon exogenous variables and dataset-provided text available at the forecast origin. These benchmark families do not constitute a controlled context ablation. Dataset details appear in Appendix A.

**Baselines.** We compare Cast-R1 with fifteen baselines across five categories. Statistical methods include ARIMA (Hyndman and Khandakar, 2008) and Prophet (Taylor and Letham, 2018). Deep learning methods include PatchTST (Nie et al., 2023), iTransformer (Liu et al., 2024b), TimeXer (Wang et al., 2024), ConvTimeNet (Cheng et al., 2025b), and DLinear (Zeng et al., 2023). Foundation models include Chronos (Ansari et al., 2025) and TimesFM (Das et al., 2024). LLM-based forecasters include OFA (Zhou et al., 2023), Time-LLM (Jin et al., 2024), TimeReasoner (Cheng et al., 2025a), and Time-R1 (Zhou et al., 2026). Time-R1 trains slow-thinking temporal reasoning via supervised warm-up and reinforcement learning. Agentic TSF frameworks include AlphaCast (Zhang et al., 2025) and TimeSeriesScientist (Zhao et al., 2025), which coordinate forecasting through LLM-driven analysis and operations. Baseline details are provided in Appendix B; EPF input access follows Appendix A.4.

**Implementation Details.** We use Qwen3-8B (Yang et al., 2025a) as the backbone and train a

| Model               | LTSF (Historical Inputs)<br>( $L = 96, H = 96$ ) |              |              |               |                 | EPF (Forecast-Origin Context)<br>( $L = 168, H = 24$ ) |                |                |               |               |
|---------------------|--------------------------------------------------|--------------|--------------|---------------|-----------------|--------------------------------------------------------|----------------|----------------|---------------|---------------|
|                     | ETTh1                                            | ETTh2        | ETTM1        | ETTM2         | Wind            | BE                                                     | DE             | FR             | NP            | PJM           |
| ARIMA               | 10.879                                           | 12.567       | 9.898        | 22.527        | 2166.64         | 869.444                                                | 362.487        | 963.997        | 55.507        | 149.281       |
| Prophet             | 46.697                                           | 23.525       | 12.225       | 31.525        | 8071.24         | 749.864                                                | 340.371        | 1024.401       | 40.754        | 53.777        |
| PatchTST            | 9.300                                            | 11.358       | 5.838        | 16.386        | 2024.83         | 512.834                                                | 201.705        | 807.512        | 29.428        | 27.164        |
| iTransformer        | <u>7.505</u>                                     | 10.161       | 5.771        | 12.751        | 1591.64         | 645.760                                                | 265.310        | 824.473        | 34.591        | 53.751        |
| ConvTimeNet         | 8.032                                            | 18.072       | 6.358        | <u>12.253</u> | 1660.13         | 668.275                                                | 198.431        | 789.035        | 29.903        | 34.638        |
| TimeXer             | 8.765                                            | 13.798       | 4.387        | 14.136        | 2063.99         | 772.704                                                | 222.798        | 803.202        | 32.052        | 34.283        |
| DLinear             | 8.506                                            | 12.347       | 5.525        | 15.272        | 2169.82         | 739.149                                                | 239.753        | <u>787.498</u> | 37.287        | 40.562        |
| Chronos             | 9.397                                            | 16.729       | 4.414        | 24.440        | 1764.15         | <u>497.075</u>                                         | 211.205        | 812.732        | 30.804        | 29.697        |
| TimesFM             | 8.096                                            | <u>9.932</u> | 5.996        | 15.232        | 1445.52         | 502.345                                                | 212.005        | 852.003        | 30.557        | 30.978        |
| OFA                 | 14.528                                           | 26.227       | 10.024       | 21.551        | 2044.332        | 621.548                                                | 275.331        | 1004.898       | 43.227        | 44.669        |
| Time-LLM            | 10.523                                           | 15.527       | 5.137        | 16.001        | 1524.75         | 523.552                                                | 232.875        | 825.667        | 35.486        | 31.998        |
| TimeReasoner        | 7.965                                            | 11.212       | 6.864        | 17.003        | 1662.55         | 549.797                                                | 286.667        | 903.977        | 46.998        | 35.778        |
| Time-R1             | 8.989                                            | 13.548       | <u>4.238</u> | 15.995        | <u>1353.938</u> | 535.372                                                | 279.146        | 880.259        | 45.765        | 34.839        |
| TimeSeriesScientist | 23.207                                           | 30.909       | 8.278        | 17.445        | 3400.916        | 906.681                                                | 245.400        | 904.655        | 46.313        | 45.400        |
| AlphaCast           | 13.395                                           | 17.350       | 6.742        | 15.185        | 1367.628        | 653.983                                                | 248.981        | 804.892        | <u>27.113</u> | <b>26.535</b> |
| <b>Cast-R1</b>      | <b>6.062</b>                                     | <b>8.405</b> | <b>3.465</b> | <b>11.181</b> | <b>1331.513</b> | <b>473.427</b>                                         | <b>183.781</b> | <b>776.898</b> | <b>24.750</b> | <u>26.905</u> |

Table 1: MSE results for two benchmark families, not a controlled context ablation. EPF provides forecast-origin context; input access follows Appendix A.4. Lower values are better. Best results are **bold**; second-best results are underlined. MAE results appear in Appendix Table 8.

| Dataset       | Length | Variables | Frequency |
|---------------|--------|-----------|-----------|
| ETTh1 & ETTh2 | 17,420 | 7         | 1 hour    |
| ETTM1 & ETTM2 | 69,680 | 7         | 15 mins   |
| Wind          | 48,673 | 7         | 15 mins   |
| BE            | 14,496 | 3         | 1 hour    |
| DE            | 14,496 | 4         | 1 hour    |
| FR            | 14,496 | 3         | 1 hour    |
| NP            | 14,496 | 3         | 1 hour    |
| PJM           | 14,496 | 3         | 1 hour    |

Table 2: Statistics of the benchmark datasets, including length, variables, and sampling frequency.

single agent on all ten datasets with four NVIDIA A800 GPUs. Trajectory distillation uses SFT on 1,200 selected teacher trajectories (120 per dataset) for 1 epoch with a learning rate of  $1.0 \times 10^{-5}$ , followed by GRPO-based RL with learning rate  $1.0 \times 10^{-6}$ , batch size 64, group size  $G = 8$ , temperature 1.0, and maximum prompt/response lengths of 8192/4096 tokens. Baselines follow official settings. All methods use matched input windows and horizons. We evaluate MSE and MAE with chronological splits of 6:2:2 (ETT) and 7:1:2 (others). We exclude future target values from the inputs, and RL rewards use only training samples. Full settings appear in Appendix D.

## 4.2 Main Results

Table 1 reports MSE results for the ten datasets. Cast-R1 achieves the lowest MSE on nine datasets and the second-lowest on PJM, where AlphaCast performs better. Averaging the per-dataset rela-

tive MSE reductions against the lowest baseline MSE, including the negative reduction on PJM, gives 8.41% overall, 12.65% for the five long-term datasets, and 4.16% for the five electricity price datasets. Compared with the agentic TSF baselines, Cast-R1 obtains lower MSE and MAE than TimeSeriesScientist on all ten datasets, and lower MSE than AlphaCast on nine and lower MAE on seven. Against the trained reasoning model Time-R1, Cast-R1 obtains lower MSE on all ten datasets and lower MAE on eight. These comparisons assess the complete framework against both agentic forecasting and forecasting-specific LLM training. Appendix Table 8 reports MAE results. Cast-R1 achieves the lowest MAE on seven datasets; AlphaCast leads on Wind and PJM, while Chronos leads on NP. The difference on Wind, where Cast-R1 has the lowest MSE but the third-lowest MAE, illustrates metric-dependent rankings. The following ablations examine how removing individual components affects the complete framework.

## 4.3 Training Ablations

### 4.3.1 Trajectory Distillation and RL Optimization

Figure 3(b) compares forecasting errors, while Figure 5(a) reports training rewards. Trajectory distillation refers to the SFT stage in Section 3.5.1. The complete pipeline has the lowest MSE on the four displayed benchmarks; removing RL yields the largest error increase, and omitting distillation

| Variant        | ETTh1        |              | ETTm1        |              | Wind            |               | DE             |              | NP            |              | PJM           |              |
|----------------|--------------|--------------|--------------|--------------|-----------------|---------------|----------------|--------------|---------------|--------------|---------------|--------------|
|                | MSE          | MAE          | MSE          | MAE          | MSE             | MAE           | MSE            | MAE          | MSE           | MAE          | MSE           | MAE          |
| w/o Features   | 7.145        | 1.988        | 4.979        | 1.548        | 1743.431        | 23.122        | 194.672        | 9.864        | 28.714        | 4.260        | 30.641        | 4.251        |
| w/o Prediction | 15.993       | 2.144        | 13.134       | 2.258        | 4047.163        | 28.113        | 564.157        | 12.965       | 64.813        | 5.267        | 91.122        | 7.135        |
| <b>Cast-R1</b> | <b>6.062</b> | <b>1.320</b> | <b>3.465</b> | <b>1.160</b> | <b>1331.513</b> | <b>16.125</b> | <b>183.781</b> | <b>7.400</b> | <b>24.750</b> | <b>3.255</b> | <b>26.905</b> | <b>3.877</b> |

Table 3: Toolkit ablations on representative datasets. Removing feature extraction or model prediction increases error. Best results are **bold**.

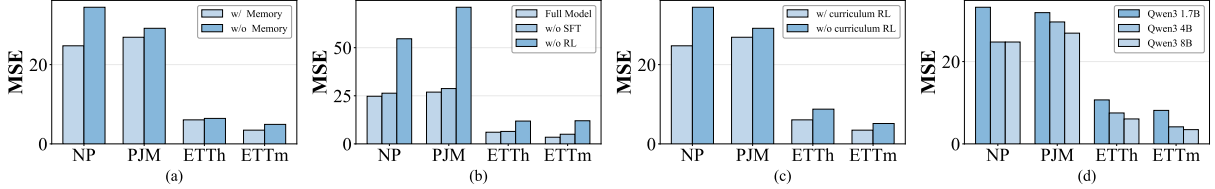


Figure 3: MSE comparisons for (a) memory-based state construction, (b) trajectory distillation (SFT) and RL, (c) curriculum learning, and (d) backbone size. ETTh/ETTm denote ETTh1/ETTm1.

| Variant        | ETTh1        |              | ETTm1        |              | Wind            |               |
|----------------|--------------|--------------|--------------|--------------|-----------------|---------------|
|                | MSE          | MAE          | MSE          | MAE          | MSE             | MAE           |
| w/o Length     | 6.102        | 1.928        | 3.982        | 1.378        | 1575.997        | 22.445        |
| w/o Error      | 13.444       | 2.832        | 12.731       | 2.681        | 3146.645        | 32.329        |
| w/o Trend      | 8.452        | 2.347        | 6.684        | 1.614        | 1830.842        | 25.219        |
| w/o Align.     | 7.847        | 2.031        | 5.984        | 1.483        | 1744.896        | 24.786        |
| <b>Cast-R1</b> | <b>6.062</b> | <b>1.320</b> | <b>3.465</b> | <b>1.160</b> | <b>1331.513</b> | <b>16.125</b> |

Table 4: Reward ablations, where Length denotes length weighting, Error measures prediction accuracy, Trend captures both trend and seasonality effects, and Align represents alignment.

(w/o SFT) also increases error. The run initialized by distillation starts with a higher reward than pure RL and generally has higher rewards during training. A possible explanation is that the teacher trajectories provide a starting policy for subsequent optimization. These observations favor combining trajectory distillation and multi-turn RL.

### 4.3.2 Reward Component Contributions

Table 4 ablates the reward groups in Equation 6: length weighting, prediction accuracy, trend and seasonality, and maximum/minimum alignment. The complete reward consistently achieves the lowest reported MSE and MAE, and removing the prediction error term yields the largest MSE increase on all three datasets. Removing the other groups also increases the reported errors, although the observed ETTh1 MSE increase without length weighting is only 0.66%. Structural terms may supply complementary optimization signals; these

error metrics do not directly measure preservation of temporal patterns or output completeness.

### 4.3.3 Curriculum-Guided Training

Figure 3(c) shows lower reported MSE with the curriculum on the four displayed benchmarks. These observed improvements favor the combination of sample filtering and progressive ordering. Exposure to increasingly difficult samples is a possible contributor, alongside changes in the retained training data. Separating filtering from ordering requires comparisons with identical retained samples and matched update budgets; the current results assess their combined effect.

## 4.4 Agent Ablations

### 4.4.1 Memory-Based State Construction

Figure 3(a) compares the complete framework with its without-memory variant. The complete framework has lower reported MSE on the four displayed benchmarks, favoring its memory-based state configuration within the evaluated workflow. The comparison with memory removed does not isolate the effect of structured state organization from retaining full interaction history. These results assess forecasting performance; they do not directly measure tool-selection correctness or reasoning reliability. The stage-aware prompt construction and memory update strategy are further described in the Appendix C.2.

**Task:** Forecast ETTh1 transformer oil temperature using 96 hourly historical observations ( $L=96$ ) over the next 96 steps ( $H=96$ ).  
**Data Characteristics:** The historical oil-temperature series contains recurring fluctuations that motivate statistical diagnosis and model routing.  
**Input Stream:** The agent uses historical observations and tool outputs; future oil-temperature targets are excluded from its decision inputs.

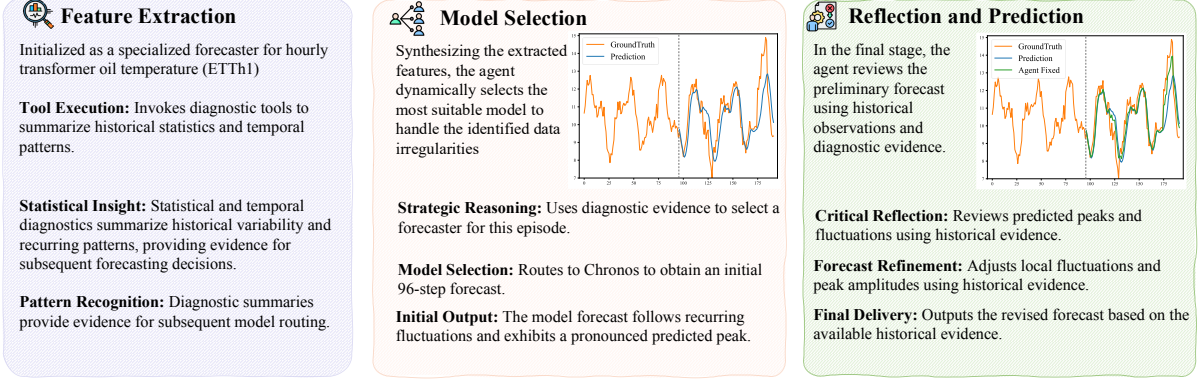


Figure 4: Qualitative ETTh1 oil-temperature forecasting case with  $L = 96$  and  $H = 96$ , illustrating statistical diagnosis, routing to Chronos, and forecast refinement. Dashed lines mark the history–forecast boundary.

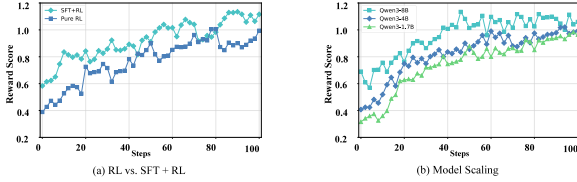


Figure 5: Training reward curves comparing (a) initialization by trajectory distillation (SFT) with pure RL and (b) different backbones.

#### 4.4.2 Diagnostic and Prediction Tools

Table 3 reports ablations of diagnostic feature extraction (w/o Features) and forecasting model invocation (w/o Prediction). Removing either component increases MSE and MAE across all six datasets compared with the full framework, with larger performance degradation generally observed when prediction tools are removed. Appendix Tables 10 and 11 further examine individual tool contributions. While most tool removals degrade forecasting performance, some yield improvements on specific datasets or metrics. These results suggest that component-level contributions are relatively consistent, whereas the effects of individual tools vary across datasets and evaluation metrics.

#### 4.5 Backbone Scaling Analysis

Figure 3(d) compares forecasting performance across Qwen3 backbone sizes, while Figure 5(b) presents their training rewards. Larger backbones achieve lower reported MSE on several benchmarks, although improvements are not uniform across all settings. The largest evaluated backbone attains the lowest or tied-lowest final MSE

and obtains higher training rewards than the smallest variant. Together, these results suggest that increased model capacity can benefit both optimization and forecasting performance within the evaluated Qwen3 family, without implying consistent gains beyond the tested configurations. Moreover, the differences across benchmarks suggest that scaling effects depend on dataset characteristics.

#### 4.6 Case Study

Figure 4 shows an ETTh1 oil-temperature forecasting episode with 96 hourly observations and a 96-step horizon. The agent uses statistical and temporal diagnostics to select Chronos based on evidence. The initial forecast captures recurring oscillations but shows local timing and amplitude errors. During reflection, the agent compares these deviations with historical evidence and adjusts selected fluctuations and peaks. This illustrates interactions among diagnostics, predictions, and reflection. Decisions rely solely on supplied inputs and tool outputs, without future oil-temperature targets. As a qualitative example, it emphasizes the decision trace rather than consistent improvement.

### 5 Conclusion

We present Cast-R1, an 8B-parameter agentic time series forecasting model trained through trajectory distillation and multi-turn GRPO with structured memory. Teacher trajectories initialize forecasting-oriented tool use, while GRPO optimizes tool selection, model routing, and forecast refinement using accuracy and temporal-structure rewards. Episode-local memory supplies diagnostic evidence and in-

intermediate forecasts to subsequent decisions. Cast-R1 achieved the lowest MSE on most of the ten benchmark datasets. Ablation studies further indicated that multi-stage training and structured memory contributed to its forecasting performance.

## Limitations

A single agent is trained on all ten datasets using Qwen3 backbones. The evaluation therefore addresses performance within these benchmark domains; transfer to unseen datasets, other domains, and other language-model families remains to be established. The harness prescribes the stage order and candidate forecasting tools, and the results do not establish open-ended workflow planning or robustness to changes in the tool pool. Memory retains evidence within each episode; cross-task experience retrieval is outside the evaluated setting.

MSE and MAE assess numerical forecasts but do not directly evaluate the faithfulness of the accompanying natural-language reasoning. The reported comparisons provide point estimates, and small differences require uncertainty estimates across independent runs. We also do not systematically measure the inference cost and latency of the multi-step agent workflow relative to single-model forecasting baselines.

## References

- Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. 2025. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*.
- Christoph Bandt and Bernd Pompe. 2002. [Permutation entropy: A natural complexity measure for time series](#). *Physical Review Letters*, 88(17):174102.
- Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. [FireAct: Toward language agent fine-tuning](#). *arXiv preprint arXiv:2310.05915*.
- Yilong Chen, Xiao Qin, Chenghao Liu, Liang Wu, Noelle I. Samia, and Kaize Ding. 2026. [LLM agents for time-series: A survey](#). *Findings of the Association for Computational Linguistics: EMNLP 2026*.
- Mingyue Cheng, Jiahao Wang, Daoyu Wang, Xiaoyu Tao, Qi Liu, and Enhong Chen. 2025a. Can slow-thinking llms reason over time? empirical studies in time series forecasting. *arXiv preprint arXiv:2505.24511*.
- Mingyue Cheng, Jiqian Yang, Tingyue Pan, Qi Liu, Zhi Li, and Shijin Wang. 2025b. ConvtimeNet: A deep hierarchical fully convolutional model for multivariate time series analysis. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 171–180.
- Mingyue Cheng, Shuo Yu, Daoyu Wang, Qingchuan Li, Xiaoyu Tao, Jie Ouyang, Yucong Luo, Yitong Zhou, Qi Liu, and Enhong Chen. 2026. [Agent-R1: A unified and modular framework for agentic reinforcement learning](#). *arXiv preprint arXiv:2511.14460*.
- Zhiqing Cui, Binwu Wang, Qingxiang Liu, Yeqiang Wang, Zhengyang Zhou, Yuxuan Liang, and Yang Wang. 2026. [Augur: Modeling covariate causal associations in time series via large language models](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 764–787. Association for Computational Linguistics.
- Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. 2024. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*.
- DeepSeek-AI. 2025. [DeepSeek-V3.2: Pushing the frontier of open large language models](#). *arXiv preprint arXiv:2512.02556*.
- Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2023. [Large language models are zero-shot time series forecasters](#). *Advances in Neural Information Processing Systems*, 36:19622–19635.
- Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. 2025. [HiAgent: Hierarchical working memory management for solving long-horizon agent tasks with large language model](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32779–32798. Association for Computational Linguistics.
- Rob J Hyndman and Yeasmin Khandakar. 2008. Automatic time series forecasting: the forecast package for R. *Journal of statistical software*, 27:1–22.
- International Energy Agency. 2025. [Electricity 2025](#). Technical report, International Energy Agency, Paris.
- Gunjan Jalori, Preetika Verma, and Sercan O Arik. 2025. [FLAIRR-TS - forecasting LLM-agents with iterative refinement and retrieval for time series](#). In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 15427–15437. Association for Computational Linguistics.
- Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. 2024. Time-llm: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*.

- Minkyong Kim, Daeun Ji, Yohan Lee, Beomsoo Kim, and Beakcheol Jang. 2026. [CTRL: Control-based time series forecasting with LLM-guided residual learning](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 21952–21968. Association for Computational Linguistics.
- Jesus Lago, Grzegorz Marcjasz, Bart De Schutter, and Rafał Weron. 2021. Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark. *Applied Energy*, 293:116983.
- Zechen Li, Baiyu Chen, Hao Xue, and Flora D. Salim. 2026. [ZARA: Training-free motion time-series reasoning via evidence-grounded LLM agents](#). *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 1:14986–15008.
- Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. 2025. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18780–18788.
- Haoxin Liu, Zhiyuan Zhao, Jindong Wang, Harshvardhan Kamarthi, and B. Aditya Prakash. 2024a. [LST-Prompt: Large language models as zero-shot time series forecasters by long-short-term prompting](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7832–7840. Association for Computational Linguistics.
- Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024b. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*.
- Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*.
- Wenzhe Niu, Zongxia Xie, Yanru Sun, Wei He, Man Xu, and Chao Hao. 2025. [LangTime: A language-guided unified model for time series forecasting with proximal policy optimization](#). In *Forty-second International Conference on Machine Learning*.
- Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. 2024. [TFB: Towards comprehensive and fair benchmarking of time series forecasting methods](#). *Proceedings of the VLDB Endowment*, 17(9):2363–2377.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- ChengAo Shen, Wenchao Yu, Fangyu Wu, Dongjin Song, Hanghang Tong, Dongsheng Luo, Wei Cheng, Haifeng Chen, and Jingchao Ni. 2026. [MetaCaster: Meta-harness-optimized agent for end-to-end few-shot learning of lightweight time series forecasters](#). *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*.
- Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. 2025. Time-moe: Billion-scale time series foundation models with mixture of experts. In *The Thirteenth International Conference on Learning Representations*.
- Xiaoyu Tao, Mingyue Cheng, Ze Guo, Shuo Yu, Yaguo Liu, Qi Liu, and Shijin Wang. 2026. [Mem-Cast: Memory-driven time series forecasting with experience-conditioned reasoning](#). *arXiv preprint arXiv:2602.03164*.
- Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician*, 72(1):37–45.
- Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. 2024. Timexer: Empowering transformers for time series forecasting with exogenous variables. *Advances in Neural Information Processing Systems*, 37:469–498.
- Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, et al. 2025. [RA-GEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning](#). *arXiv preprint arXiv:2504.20073*.
- Muyan Weng, Defu Cao, Wei Yang, Yashaswi Sharma, and Yan Liu. 2026. [TemporalBench: A benchmark for evaluating LLM-based agents on contextual and event-informed time series tasks](#). *arXiv preprint arXiv:2602.13272*.
- Xingjian Wu, Junkai Lu, Zhengyu Li, Xiangfei Qiu, Jilin Hu, Chenjuan Guo, Christian S Jensen, and Bin Yang. 2026. Timeart: Towards agentic time series reasoning via tool-augmentation. *arXiv preprint arXiv:2601.13653*.
- Xiaogang Xu, Jiaqi Tang, Jianmin Chen, Yingying Yan, Zhenchao Tang, Xiangxin Zhou, Xiaobin Hu, Wei Wei, Jinfeng Wu, Qifeng Chen, Lu Zhou, Jiafei Wu, Zhe Liu, Jianwei Yin, and Weimin Zheng. 2026. [LLM-based agents for forecasting and prediction: Methods, training, evaluation, and applications](#). *arXiv preprint arXiv:2608.23058*.
- Hao Xue and Flora D Salim. 2024. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6851–6864.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

- Silin Yang, Dong Wang, Haoqi Zheng, and Ruochun Jin. 2025b. [TimeRAG: Boosting LLM time series forecasting via retrieval-augmented generation](#). In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.
- Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128.
- Xiaohan Zhang, Tian Gao, Mingyue Cheng, Bokai Pan, Ze Guo, Yaguo Liu, and Xiaoyu Tao. 2025. Alphacast: A human wisdom-llm intelligence co-reasoning framework for interactive time series forecasting. *arXiv preprint arXiv:2511.08947*.
- Haokun Zhao, Xiang Zhang, Jiaqi Wei, Yiwei Xu, Yuting He, Siqi Sun, and Chenyu You. 2025. Timeseries-scientist: A general-purpose ai agent for time series analysis. *arXiv preprint arXiv:2510.01538*.
- Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115.
- Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. 2023. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355.
- Yitong Zhou, Yucong Luo, Mingyue Cheng, Qi Liu, Jiahao Wang, Daoyu Wang, and Enhong Chen. 2026. [Time series forecasting via reasoning: A slow-thinking approach with reinforcement fine-tuned llms](#). In *Proceedings of the 35th ACM International Conference on Information and Knowledge Management*.

## A Datasets and Evaluation Protocol

We evaluate ten datasets: five short-term electricity price forecasting (EPF) benchmarks and five long-term time series forecasting (LTSF) benchmarks. Each task predicts a single target series from historical observations and available covariates. Table 5 lists the targets and original covariate fields; the following subsections describe the datasets, task configuration, input access, and evaluation metrics.

### A.1 Electricity Price Forecasting

The EPF benchmarks (Lago et al., 2021) contain hourly day-ahead electricity prices from five markets: Nordic Nord Pool (NP), the Commonwealth Edison (ComEd) zone of PJM, Belgium (BE), France (FR), and Germany (DE). Their load and generation forecast fields are listed in Table 5; the German load forecast covers the Amprion area. EPF additionally supplies forecast-horizon covariates and dataset-provided text available at the forecast origin, following the input-access protocol in Appendix A.4.

### A.2 Long-term Forecasting

**ETT.** The Electricity Transformer Temperature benchmark (Zhou et al., 2021) records oil temperature (OT) and six load variables (HUFL, HULL, MUFL, MULL, LUFL, and LULL) over two years. OT is the target forecasting variable in this study. ETTh1 and ETTh2 use hourly sampling; ETTm1 and ETTm2 use 15-minute sampling.

**Wind.** The Wind dataset from TFB (Qiu et al., 2024) contains 48,673 observations at 15-minute intervals and seven variables. Wind power is the target; the accompanying fields are forecasts of wind speed, wind direction, temperature, pressure, and humidity, plus measured wind speed. These fields are used only within the historical window.

### A.3 Forecasting Task Configuration

The chronological train/validation/test split is 6:2:2 for ETT and 7:1:2 for the other datasets. RL rewards use training-split targets only. Table 6 gives the matched input windows and horizons. A horizon of 96 steps corresponds to 96 hours for ETTh and 24 hours for the 15-minute ETTm and Wind series.

### A.4 Input Access

All methods receive historical target observations. Covariate-capable methods receive the same

dataset-provided variables within the historical window. LTSF uses these historical inputs. For EPF, Cast-R1 additionally uses forecast-horizon exogenous covariates and dataset-provided textual context available at the forecast origin. Baselines use these additional inputs when supported by their input interfaces; they do not consume unsupported modalities. Future target values are excluded from the inputs in both benchmark families. Tables 1 and 8 therefore compare complete forecasting systems under their supported input interfaces. Matched windows and horizons do not imply identical input modalities, and these comparisons do not isolate the effect of additional context.

### A.5 Evaluation Metrics

We report mean squared error (MSE) and mean absolute error (MAE) on the raw data scale:

$$\begin{aligned} \text{MSE} &= \frac{1}{H} \sum_{i=1}^H (y_i - \hat{y}_i)^2, \\ \text{MAE} &= \frac{1}{H} \sum_{i=1}^H |y_i - \hat{y}_i|. \end{aligned} \quad (7)$$

Here,  $y_i$  and  $\hat{y}_i$  are the target and prediction at horizon step  $i$ ; these equations define the errors for one forecasting episode. We evaluate forecasting accuracy over the complete prediction horizon, capturing both large deviations through MSE and overall prediction discrepancies through MAE. Lower values indicate better forecasting performance under both metrics.

**Relative MSE Reduction.** The improvement reported in the main text averages relative reductions within datasets. Let  $b_d$  be the lowest baseline MSE and  $c_d$  the Cast-R1 MSE on dataset  $d$ . For a dataset group  $\mathcal{D}$ , the reported percentage is:

$$\Delta_{\mathcal{D}} = \frac{100}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \frac{b_d - c_d}{b_d}. \quad (8)$$

Each dataset contributes equally, and negative reductions are retained. This gives 8.41% across all ten datasets, 12.65% for LTSF, and 4.16% for EPF. The calculation uses relative errors because raw-scale MSEs have different units across datasets. Accordingly, the reported gain should be interpreted as the average percentage improvement across datasets, rather than the reduction obtained by pooling squared errors over all available individual forecasting test observations.

Table 5: Forecast targets and original covariate fields. EPF additionally uses forecast-horizon covariates and textual context available at the forecast origin. LTSF uses covariates only within the historical window. Textual context is additional to the listed fields.

| Dataset     | Target            | Freq.  | Task | Covariate fields                                              |
|-------------|-------------------|--------|------|---------------------------------------------------------------|
| NP          | Electricity price | 1 h    | EPF  | Grid load and wind generation forecasts.                      |
| PJM         | Electricity price | 1 h    | EPF  | System-wide and zonal load forecasts.                         |
| BE          | Electricity price | 1 h    | EPF  | National load and French generation forecasts.                |
| FR          | Electricity price | 1 h    | EPF  | Grid load and domestic generation forecasts.                  |
| DE          | Electricity price | 1 h    | EPF  | Zonal load, wind, and solar generation forecasts.             |
| ETTh1/ETTh2 | Oil temperature   | 1 h    | LTSF | Six power load variables accompanying oil temperature.        |
| ETTm1/ETTm2 | Oil temperature   | 15 min | LTSF | The same six power load variables as the hourly ETT variants. |
| Wind        | Wind power        | 15 min | LTSF | Weather forecasts and measured wind speed.                    |

Table 6: Forecasting task configuration.  $L$  and  $H$  count time steps; duration gives the physical length of the forecasting horizon. Split denotes chronological train/validation/test proportions.

| Family      | $L$ | $H$ | Duration | Split |
|-------------|-----|-----|----------|-------|
| EPF         | 168 | 24  | 24 hours | 7:1:2 |
| ETTh1/ETTh2 | 96  | 96  | 96 hours | 6:2:2 |
| ETTm1/ETTm2 | 96  | 96  | 24 hours | 6:2:2 |
| Wind        | 96  | 96  | 24 hours | 7:1:2 |

## B Baselines

We compare Cast-R1 with fifteen baselines in five categories. Shared evaluation settings appear in Appendix A.3, and implementation settings are summarized below in Appendix B.2.

### B.1 Baseline Methods

#### Statistical Methods.

- **ARIMA** (Hyndman and Khandakar, 2008): autoregression, differencing, and moving averages for statistical extrapolation.
- **Prophet** (Taylor and Letham, 2018): an additive model of trend, seasonality, and holiday effects.

#### Deep Learning Methods.

- **DLinear** (Zeng et al., 2023): linear forecasting with trend and seasonal decomposition.
- **PatchTST** (Nie et al., 2023): channel-independent temporal modeling with patch-based Transformer inputs.
- **iTransformer** (Liu et al., 2024b): attention across variate tokens to model cross-variable dependencies.
- **TimeXer** (Wang et al., 2024): forecasting that incorporates and aligns exogenous variables.
- **ConvTimeNet** (Cheng et al., 2025b): hierarchical

cal convolutional forecasting with adaptive segmentation and deformable patching.

#### Foundation Models.

- **Chronos** (Ansari et al., 2025): pretrained univariate, multivariate, and covariate-informed forecasting with group attention.
- **TimesFM** (Das et al., 2024): a pretrained decoder-only Transformer using temporal patches for zero-shot forecasting.

#### LLM-based Methods.

- **OFA** (Zhou et al., 2023): adapts a pretrained language model by tuning selected layers while retaining frozen attention and feedforward layers.
- **Time-LLM** (Jin et al., 2024): reprograms time series into language-model representations with prompt-based conditioning.
- **TimeReasoner** (Cheng et al., 2025a): uses LLM reasoning to assess temporal dynamics and produce forecasts.
- **Time-R1** (Zhou et al., 2026): combines supervised fine-tuning warmup with reinforcement learning for slow-thinking, multi-step temporal reasoning.

#### Agentic TSF Methods.

- **AlphaCast** (Zhang et al., 2025): a training-free workflow combining temporal feature reasoning, knowledge and context, and reflection across multiple stages.
- **TimeSeriesScientist** (Zhao et al., 2025): coordinates specialized agents for diagnostics and preprocessing, model planning, fitting, validation and ensembling, and reporting.

## B.2 Baseline Implementation

Deep learning baselines use their official implementations and Adam with a learning rate of  $10^{-3}$  or  $10^{-4}$  according to their recommended configurations. Early stopping uses patience 3. Inputs use Z-score standardization (zero mean and unit variance), and forecasts are returned to the raw scale for evaluation.

## C Agent Implementation

This section defines the diagnostic statistics, episode memory, and stage-aware prompt construction used by the forecasting harness.

### C.1 Diagnostic Tool Statistics

The toolkit in Section 3.3 uses the following representative statistics. For a historical channel  $v_{1:L}$ , let  $I$  denote its observed indices and  $n = |I|$ .

**Data Quality Assessment.** These tools identify missing values, saturation effects, and constant or abnormal channels. The missing-value ratio is  $q_{\text{miss}} = 1 - \frac{n}{L}$ , helping distinguish temporal patterns from potential data issues.

**Global Statistical Characterization.** Global summaries describe central tendency, dispersion, correlation, and frequency-domain properties. Variability is summarized by the mean absolute deviation  $D_{\text{abs}} = \frac{1}{n} \sum_{i \in I} |v_i - \mu_v|$ , where  $\mu_v$  is the mean of observed values and  $n > 0$ .

**Structural and Dynamic Analysis.** Within-series tools detect change points, stable segments, and local extrema. For a fully observed segment  $[a, b]$ , the average slope is  $\text{slope}_{[a,b]} = \frac{v_b - v_a}{b - a}$ , where  $a < b$ , providing evidence about trends and regime shifts.

**Event-level Summarization.** Event tools summarize rising, declining, stable, and oscillatory behavior. Pattern prevalence is  $\nu_c = \frac{n_c}{S}$ , where  $n_c$  counts segments labeled  $c$  and  $S > 0$  is the total number of segments. The pattern with the largest prevalence summarizes the dominant behavior.

**Model Adequacy and Residual Diagnostics.** Residual tools assess how well a simple baseline explains history using  $r_i = v_i - \check{v}_i$ , where  $\check{v}_i$  is a baseline fitted value on history. Residual distributions, autocorrelation, and tail behavior guide modeling and refinement decisions.

## C.2 Memory Update and State Assembly

For each episode, memory  $M_k = (D_k, Q_k)$  stores **Analysis History**  $D_k$ : diagnostic summaries, structural indicators, and event descriptions; and **Prediction Results**  $Q_k$ : forecasts returned by model tools. Both components start empty and retain the current episode’s tool results across turns. The environment manages this memory and constructs the next state as in Equation 1. During RL training, multi-turn GRPO updates the shared decision policy. Historical observations and task context are supplied separately from these two memory components.

At each decision step,  $P$  constructs a new policy-visible prompt  $s_k = P(z, M_k)$  from task inputs and structured memory. The available diagnostics and forecasts determine the stage-specific instructions and permitted actions. Previous agent responses and raw tool messages are not additionally appended as conversation history; tool-derived evidence is carried forward through  $D_k$  and  $Q_k$ . Early prompts contain the full historical window and task context. Refinement prompts combine recent observations, task context, and retained diagnostics and forecasts. EPF context contains forecast-horizon exogenous covariates and textual information available at the forecast origin; LTSF uses historical inputs.

### C.3 Stage-aware Prompts

- **Preparation:** when analysis is absent, supply the full historical window and task context and permit feature extraction and diagnostic tools. Independent tool calls may execute in parallel within one response.
- **Forecasting:** when analysis exists but no forecast is stored, supply historical inputs, task context, and Analysis History and permit forecasting model invocation with evidence-guided model selection.
- **Refinement and Output:** when diagnostics and forecasts are available, supply recent observations, task context, Analysis History, and Prediction Results and permit reasoning, refinement, and final output.

These labels identify logical stages rather than a fixed number of agent responses. Episodes follow the decision budget  $K_{\text{max}}$  defined in the main text. The environment constrains the stage order and permitted actions, while the policy selects tools,

routes models, and revises predictions within this protocol.

**Nominal Episode Procedure.** The state transitions in Equation 1 can be summarized as follows:

1. Initialize  $M_1 = (\emptyset, \emptyset)$  for the supplied forecasting input  $z$ .
2. At decision step  $k \leq K_{\max}$ , construct  $s_k = P(z, M_k)$  and determine the permitted actions from the available diagnostics and forecasts.
3. Generate  $a_k \sim \pi_\theta(\cdot | s_k)$ . If it provides the final forecast, terminate the episode with  $\hat{y}(\tau)$ .
4. Otherwise execute its tool calls, obtain  $o_k = T(s_k, a_k)$ , update  $M_{k+1} = U(M_k, a_k, o_k)$ , and continue with the next state.

During RL, the final forecast is scored against the training target  $y$ . At inference, the agent completes the procedure using the supplied inputs and tool outputs. This summary describes the nominal interaction; runtime budgets and failure handling are separate implementation settings.

#### C.4 Prompt Template

Box 1 summarizes the stage protocol and representative tool calls. Inputs follow Appendices A.3 and A.4. XML-style tags separate reflection from serialized timestamp–value predictions. This summary illustrates the workflow rather than reproducing a verbatim run prompt.

Negative wholesale electricity prices can be valid market outcomes (International Energy Agency, 2025); the prompt therefore requires contextual evidence before revising them.

#### Box 1: Forecasting Prompt Summary

**Role & Inputs:** Forecast the requested target and horizon from historical inputs. For EPF, also use forecast-horizon covariates and text available at the forecast origin.

**Preparation:** Assess data quality, distribution, dynamics, and events with diagnostic tools such as `extract_basic_statistics` and `extract_within_channel_dynamics`. Store outputs in Analysis History. Use diagnostic tools only at this stage.

**Forecasting:** Use Analysis History to select ARIMA, Chronos, PatchTST, or iTransformer via `predict_time_series`. Store the forecast in Prediction Results.

**Refinement:** Use recent observations, task context, and both memory components to review and revise anomalous patterns. Negative electricity prices alone do not justify revision.

**Final Output:** Use `<think>...</think>` for reflection and `<answer>...</answer>` for timestamp–value predictions; return only these two blocks.

## D Training Details

We describe the computational setup, trajectory distillation, multi-turn RL, curriculum, and reward computation. Task and evaluation settings appear in Appendix A.3; baseline settings appear in Appendix B.2.

### D.1 Model and Computational Environment

We train a single Qwen3-8B agent on samples from all ten datasets using PyTorch and Hugging Face Transformers. Both SFT and RL run on four NVIDIA A800 GPUs with 80GB memory per device. RL uses gradient checkpointing and BF16 precision for efficient large-scale training. Table 7 summarizes SFT and GRPO settings.

Table 7: Hyperparameter settings for Cast-R1 training.

| Hyperparameter             | SFT                  | GRPO                 |
|----------------------------|----------------------|----------------------|
| Base Model                 | Qwen3-8B             |                      |
| Precision                  | BF16                 |                      |
| Optimizer                  | AdamW                | AdamW                |
| Learning Rate              | $1.0 \times 10^{-5}$ | $1.0 \times 10^{-6}$ |
| LR Scheduler               | Cosine               | Cosine               |
| Epochs                     | 1                    | 1                    |
| Global Batch Size          | 16                   | 64                   |
| Group Size ( $G$ )         | -                    | 8                    |
| Max Prompt Length          | 4096                 | 8192                 |
| Max Response Length        | 2048                 | 4096                 |
| Temperature                | -                    | 1.0                  |
| KL Coefficient ( $\beta$ ) | -                    | 0.04                 |
| Gradient Accumulation      | 1                    | 4                    |

### D.2 Trajectory Distillation

**Teacher Demonstrations.** A DeepSeek-V3.2 teacher generates multi-turn trajectories using the forecasting agent harness, with forecasting windows drawn exclusively from the training splits of the ten datasets. The trajectories cover diagnostic tool invocation, forecasting model selection, and prediction refinement.

#### D.2.1 Trajectory Evaluation and Selection

We apply a two-stage filtering procedure to the teacher-generated trajectories.

**Execution Validity.** We first check whether each trajectory completes the required tool interactions and produces a valid forecast. Trajectories with failed tool execution, malformed outputs, or non-finite prediction values are discarded.

**Forecasting Quality.** For each valid trajectory  $\tau$ , we evaluate its final prediction against the ground-truth target from the training split. For forecasting

horizon  $H$ , the error is:

$$E(\tau) = \frac{1}{H} \sum_{h=1}^H (\hat{y}_h(\tau) - y_h)^2. \quad (9)$$

Within each dataset, valid trajectories are ranked by ascending forecasting MSE. We retain the 120 lowest-error complete trajectories per dataset, yielding 1,200 demonstrations with equal dataset representation in  $D_{\text{SFT}}$ .

Trajectory selection evaluates completed teacher interactions. The curriculum instead scores forecasting inputs to determine sample difficulty and training order (Appendix D.4).

### D.2.2 Supervised Distillation

Each retained demonstration pairs a teacher response with its decision state; accumulated tool feedback supplies context for subsequent responses. The supervised targets in Equation 3 are the teacher responses containing tool calls or the final forecast. The Qwen3-8B student is fine-tuned for one epoch with AdamW before RL. For a retained trajectory with  $K(\tau^*)$  responses, the corresponding supervision contains  $K(\tau^*)$  state-response pairs. Thus, the response-example count is:

$$N_{\text{resp}} = \sum_{\tau^* \in D_{\text{SFT}}} K(\tau^*). \quad (10)$$

This distinguishes the number of complete demonstrations from the number of supervised responses. In autoregressive form,

$$\begin{aligned} & \log \pi_{\theta}(a_k^* | s_k^*) \\ &= \sum_{j=1}^{|a_k^*|} \log \pi_{\theta}(a_{k,j}^* | s_k^*, a_{k,<j}^*). \end{aligned} \quad (11)$$

The state contains the task inputs and available tool observations, while the response tokens supply the likelihood terms in Equation 3.

### D.3 Multi-turn Reinforcement Learning

Multi-turn GRPO samples  $G = 8$  trajectories per input and computes group-relative advantages from terminal rewards. Sampling uses temperature 1.0. The shared training configuration is listed in Table 7. For trajectory  $i$ , response  $k$ , and response token  $j$ , let  $h_{i,k,j} = (s_{i,k}, a_{i,k,<j})$ . The token-level expansion of the outcome-supervised GRPO formulation (Shao et al., 2024) uses:

$$\rho_{i,k,j}(\theta) = \frac{\pi_{\theta}(a_{i,k,j} | h_{i,k,j})}{\pi_{\text{old}}(a_{i,k,j} | h_{i,k,j})} \quad (12)$$

and the clipped surrogate:

$$\ell_{i,k,j}(\theta) = \min\{\rho_{i,k,j}(\theta) \hat{A}_i, \text{clip}(\rho_{i,k,j}(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_i\}, \quad (13)$$

where  $\epsilon$  is the clipping parameter and  $\hat{A}_i$  is the trajectory advantage in Equation 5. The same outcome advantage applies across the trajectory’s agent responses, with KL regularization against a reference policy. Tool outputs condition the next state as environment observations. Rewards within a group are compared for the same input and target; they are not pooled across unrelated tasks.

### D.4 Curriculum Learning

The curriculum combines the prediction MSE of a reference forecaster with normalized permutation entropy (Bandt and Pompe, 2002) of the historical series. Prediction error measures difficulty relative to that forecaster, while permutation entropy characterizes the series’ intrinsic complexity. These signals guide filtering of noise-dominated samples and the progressive introduction of retained samples from simpler trends and seasonal patterns to more complex dynamics. The reference forecaster used for difficulty scoring has a separate role from the DeepSeek-V3.2 teacher that generates interaction trajectories. The trajectory selection procedure in Appendix D.2.1 determines which completed teacher interactions provide SFT supervision, whereas the curriculum operates on forecasting inputs to determine sample difficulty and training order.

For a scalar sequence  $v_{1:L}$ , embedding order  $m \geq 2$ , and delay  $\ell \geq 1$  with  $L > (m - 1)\ell$ , ordinal patterns are obtained from vectors  $(v_t, v_{t+\ell}, \dots, v_{t+(m-1)\ell})$ . Let  $p_{\sigma}$  be the frequency of pattern  $\sigma$  among the  $m!$  possible orderings in  $\Pi_m$ . Normalized permutation entropy is:

$$\text{PE}_{m,\ell}(v) = -\frac{1}{\log(m!)} \sum_{\sigma \in \Pi_m} p_{\sigma} \log p_{\sigma}, \quad (14)$$

with  $0 \log 0 = 0$  and  $\text{PE}_{m,\ell} \in [0, 1]$ . Higher values indicate a more uniform distribution of local ordinal patterns. This statistic measures a different property from prediction error, which depends on the reference forecaster.

### D.5 Reward Implementation

**Output Format.** Both `<answer>` and `</answer>` are required. If either tag is missing, reward computation stops and returns  $-1$ . The tags act as a

| Model               | LTSF (Historical Inputs)<br>( $L = 96, H = 96$ ) |              |              |              |               | EPF (Forecast-Origin Context)<br>( $L = 168, H = 24$ ) |              |              |              |              |
|---------------------|--------------------------------------------------|--------------|--------------|--------------|---------------|--------------------------------------------------------|--------------|--------------|--------------|--------------|
|                     | ETTh1                                            | ETTh2        | ETTm1        | ETTm2        | Wind          | BE                                                     | DE           | FR           | NP           | PJM          |
| ARIMA               | 2.429                                            | 3.248        | 2.513        | 4.325        | 28.767        | 13.549                                                 | 11.150       | 13.224       | 4.116        | 8.481        |
| Prophet             | 4.592                                            | 3.548        | 2.504        | 3.781        | 56.059        | 14.988                                                 | 12.238       | 14.422       | 3.941        | 5.739        |
| PatchTST            | 1.638                                            | 2.008        | 1.397        | 1.945        | 20.465        | <u>10.400</u>                                          | <u>8.029</u> | <u>8.287</u> | 3.474        | 4.103        |
| iTransformer        | 1.513                                            | <u>1.930</u> | 1.460        | <u>1.661</u> | 18.186        | 14.375                                                 | 11.230       | 12.618       | 4.039        | 5.501        |
| ConvTimeNet         | <u>1.439</u>                                     | 2.578        | <u>1.252</u> | 2.057        | 18.343        | 10.804                                                 | 9.405        | 8.993        | 3.365        | 4.313        |
| TimeXer             | 2.204                                            | 2.663        | 1.609        | 3.113        | 31.914        | 12.286                                                 | 10.274       | 9.483        | 3.490        | 4.252        |
| DLinear             | 2.239                                            | 2.558        | 1.751        | 2.509        | 29.325        | 12.590                                                 | 10.619       | 9.735        | 3.896        | 4.508        |
| Chronos             | 2.250                                            | 2.820        | 1.323        | 2.117        | 17.182        | 11.129                                                 | 8.584        | 8.423        | <b>3.155</b> | <u>3.712</u> |
| TimesFM             | 2.223                                            | 2.231        | 2.012        | 2.114        | 20.898        | 11.284                                                 | 10.032       | 9.932        | 4.667        | 5.001        |
| OFA                 | 2.917                                            | 3.895        | 2.771        | 3.152        | 26.557        | 13.665                                                 | 14.235       | 13.505       | 4.462        | 4.913        |
| Time-LLM            | 2.500                                            | 2.979        | 1.668        | 3.352        | 18.775        | 10.997                                                 | 9.927        | 10.975       | 5.221        | 4.139        |
| TimeReasoner        | 1.945                                            | 2.551        | 2.552        | 2.925        | 22.559        | 11.976                                                 | 15.223       | 15.997       | 6.661        | 4.312        |
| Time-R1             | 2.136                                            | 2.599        | 1.376        | 3.351        | <u>15.110</u> | 10.710                                                 | 13.614       | 14.307       | 5.957        | 3.856        |
| TimeSeriesScientist | 3.022                                            | 3.972        | 1.921        | 3.949        | 25.648        | 14.939                                                 | 10.388       | 12.71        | 4.156        | 5.388        |
| AlphaCast           | 2.345                                            | 2.924        | 1.733        | 3.566        | <b>13.065</b> | 10.713                                                 | 10.228       | 9.103        | <u>3.191</u> | <b>3.586</b> |
| <b>Cast-R1</b>      | <b>1.320</b>                                     | <b>1.747</b> | <b>1.160</b> | <b>1.581</b> | 16.125        | <b>10.332</b>                                          | <b>7.400</b> | <b>8.105</b> | 3.255        | 3.877        |

Table 8: MAE results for the same benchmark families and input regimes as Table 1. The groups do not constitute a controlled context ablation; input access follows Appendix A.4. Lower values are better. Best values are **bold**; second-best values are underlined.

validity gate; outputs passing this check are scored by Equation 6.

**Accuracy and Temporal Structure.** Prediction MSE uses predictions and targets normalized by the same target mean and standard deviation. This target-dependent normalization is used only in training reward computation; the reported MSE and MAE use the raw scale. Each nonnegative prediction, seasonal, or trend error  $e$  is mapped to  $q(e) = [1 + \log(1 + e)]^{-1}$ . Moving averages decompose both sequences into trend and seasonal components. Failed decomposition contributes zero for the affected component. Local maxima and minima are detected with a sliding window and matched within  $\pm 2$  time steps; their hit rates supply the alignment scores. For aligned  $H$ -step sequences and a positive target standard deviation  $\sigma_y$ , the common normalization gives

$$e_{\text{pred}} = \frac{1}{H\sigma_y^2} \sum_{i=1}^H (\hat{y}_i - y_i)^2. \quad (15)$$

Consequently, a target and its forecast are compared on a common scale; they are not standardized independently. The score mapping has  $q(0) = 1$  and decreases with error. The training reward and the raw-scale evaluation metrics therefore serve distinct roles.

**Weights and Completeness.** Table 9 lists the terms in Equation 6. The length factor  $p =$

$\min(1, n_{\text{pred}}/n_{\text{gt}})$  scales the entire weighted sum, including the constant  $w_{\text{len}}$ . It penalizes outputs shorter than the target and saturates at one for longer outputs; by itself, it does not enforce exactly  $H$  predictions. For component scores  $u_j \in [0, 1]$ , the weights give  $0 \leq R_{\text{valid}} \leq 1.2p \leq 1.2$ .

Table 9: Components of the valid-output reward before multiplication by the length factor  $p$ .

| Component           | Score                 | Weight |
|---------------------|-----------------------|--------|
| Length constant     | 1                     | 0.10   |
| Prediction accuracy | $q(e_{\text{pred}})$  | 0.70   |
| Seasonality         | $q(e_{\text{seas}})$  | 0.05   |
| Trend               | $q(e_{\text{trend}})$ | 0.15   |
| Maximum alignment   | Maximum hit rate      | 0.10   |
| Minimum alignment   | Minimum hit rate      | 0.10   |

## E Additional Experimental Results

We first report the full MAE comparison, then examine diagnostic tools, forecasting models, planning, and refinement. A qualitative forecast comparison complements these aggregate results.

### E.1 MAE Results and Standalone Predictor Comparison

Table 8 complements the MSE comparison in Table 1. Cast-R1 achieves the lowest MAE on seven datasets. AlphaCast achieves the lowest MAE on Wind and PJM, while Chronos leads on NP. On Wind, Cast-R1 ranks first in MSE and third in MAE, showing that method rankings differ across

the two error criteria. Comparing Cast-R1 with the best individual baseline among the four candidate forecasters (ARIMA, Chronos, PatchTST, and iTransformer) gives lower MSE on all ten datasets. The per-dataset reductions range from 0.95% to 21.50%, averaging 12.09%. These numbers describe the complete agent-and-tool pipeline relative to standalone predictors; they do not isolate the contribution of model routing from refinement.

## E.2 Diagnostic Tool Ablations

Table 10 compares diagnostic removals on ETTh1 and NP. Each variant removes the corresponding diagnostic group defined in Section 3.3 and Appendix C.1. Removing basic statistics causes the largest reported MSE increase on both datasets. All diagnostic removals worsen ETTh1 results. On NP, removing data quality assessment reduces MSE by 4.82% and MAE by 0.86%, while removing residual diagnostics or channel dynamics improves MSE only. These comparisons show dataset- and metric-dependent contributions of individual diagnostic groups.

Table 10: Diagnostic tool ablations on ETTh1 and NP.

| Variant                  | ETTh1        |              | NP            |              |
|--------------------------|--------------|--------------|---------------|--------------|
|                          | MSE          | MAE          | MSE           | MAE          |
| w/o Basic Stats          | 9.814        | 2.384        | 36.781        | 4.715        |
| w/o Channel Dynamics     | 7.012        | 1.532        | 24.711        | 3.386        |
| w/o Residual Diagnostics | 6.543        | 1.341        | 23.774        | 3.335        |
| w/o Data Quality         | 6.942        | 1.574        | <b>23.557</b> | <b>3.227</b> |
| w/o Event Patterns       | 7.571        | 1.642        | 27.154        | 3.646        |
| <b>Cast-R1</b>           | <b>6.062</b> | <b>1.320</b> | 24.750        | 3.255        |

## E.3 Forecasting Model Ablations

Table 11 compares individual forecasting model removals on ETTh1 and NP. Each variant removes the named forecaster from the candidate pool  $B$  in Section 3.3. Every tested removal increases MSE and MAE on both datasets. Removing Chronos produces the largest reported MSE increase on NP, whereas removing PatchTST has the strongest effect on ETTh1. These results show dataset-dependent sensitivity to the available forecasters and favor retaining the full candidate pool in the two evaluated settings.

## E.4 Planning and Refinement Ablations

Table 12 compares planning and refinement removals on ETTh1 and NP. Planning refers to diagnostic tool selection and evidence-guided model

Table 11: Forecasting model ablations on ETTh1 and NP.

| Variant          | ETTh1        |              | NP            |              |
|------------------|--------------|--------------|---------------|--------------|
|                  | MSE          | MAE          | MSE           | MAE          |
| w/o ARIMA        | 6.614        | 1.413        | 24.788        | 3.334        |
| w/o Chronos      | 8.414        | 1.622        | 55.418        | 7.097        |
| w/o PatchTST     | 14.261       | 3.014        | 37.191        | 4.277        |
| w/o iTransformer | 13.512       | 2.825        | 30.387        | 4.054        |
| <b>Cast-R1</b>   | <b>6.062</b> | <b>1.320</b> | <b>24.750</b> | <b>3.255</b> |

routing, while refinement refers to numerical revision of the intermediate forecast, following the workflow in Section 3. These are component ablations within the prescribed stage order in Appendix C.3. Removing either component increases MSE and MAE, with planning removal causing the larger deterioration on both datasets. Relative to the variant without refinement, the full framework reduces MSE by 33.69% on ETTh1 and 2.61% on NP. The reported gains therefore differ in magnitude across these two datasets.

Table 12: Agent component ablations on ETTh1 and NP, removing planning or refinement.

| Variant        | ETTh1        |              | NP            |              |
|----------------|--------------|--------------|---------------|--------------|
|                | MSE          | MAE          | MSE           | MAE          |
| w/o Planning   | 18.184       | 3.914        | 38.540        | 5.549        |
| w/o Refinement | 9.142        | 1.542        | 25.413        | 3.841        |
| <b>Cast-R1</b> | <b>6.062</b> | <b>1.320</b> | <b>24.750</b> | <b>3.255</b> |

## E.5 Forecast Visualization

Figure 6 compares Cast-R1 with selected baseline forecasters on an ETTh1 oil-temperature segment with  $L = 96$  and  $H = 96$ , recurring oscillations, and a sharp downward excursion. Cast-R1 retains the oscillatory pattern and tracks several later peaks, although it does not fully capture the deepest drop and some peak amplitudes. Other forecasters differ in timing, level, and amplitude, with several curves smoothing peaks or shifting troughs. This example illustrates local tracking behavior and remaining extreme-event errors; aggregate performance is reported in Tables 1 and 8.

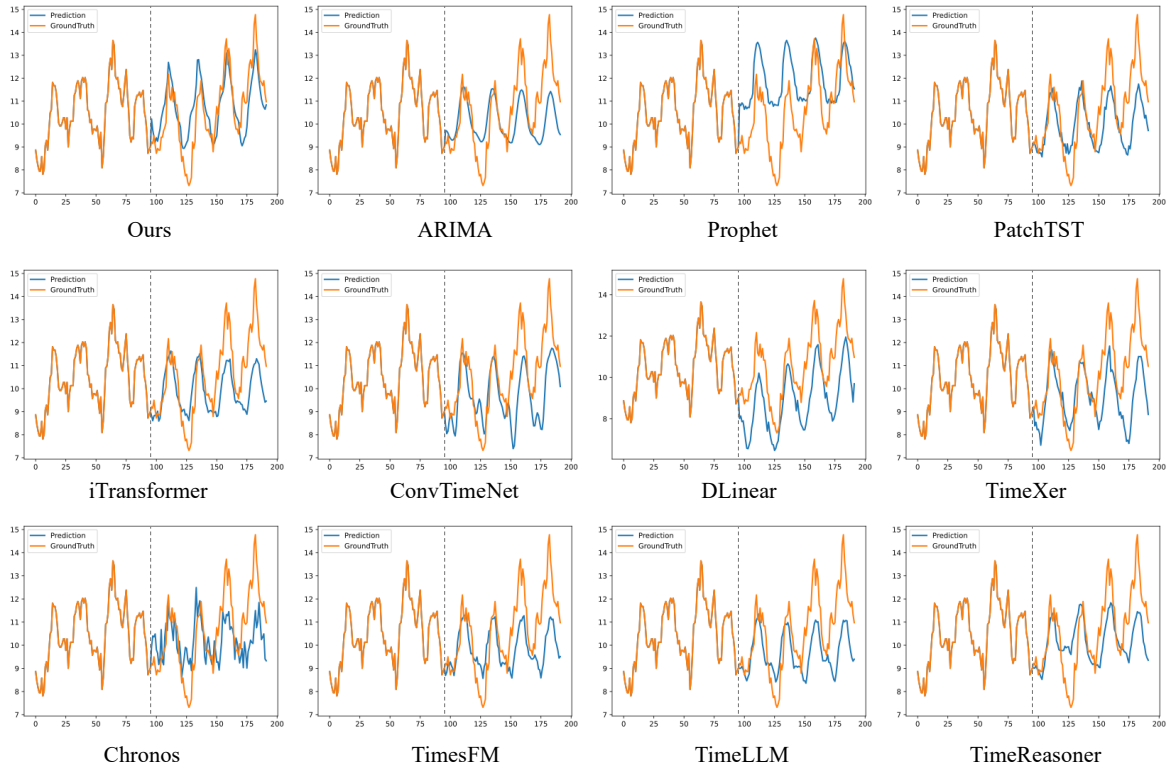


Figure 6: Forecast comparison on ETTh1 oil temperature with  $L = 96$  and  $H = 96$ . Dashed lines separate the 96 historical observations from the 96-step forecasting horizon.