The VLDB Journal manuscript No.
(will be inserted by the editor)

CastFlow: Bridging Evidence Acquisition and Accurate Prediction in Agentic Time Series Forecasting

Bokai Pan · **Mingyue Cheng** · **Xiaoyu Tao** · **Zhiding Liu** · **Shuo Yu** · **Yuchong Wu** · **Qi Liu** · **Defu Lian** · **Enhong Chen**

Received: date / Accepted: date

Abstract Time series forecasting is a fundamental task in temporal data analytics, and large language model (LLM)-based agents offer a promising paradigm through autonomous tool interaction and evidence-grounded reasoning. However, two fundamental challenges remain: how to acquire forecast-relevant evidence through adaptive tool interactions, and how to translate such evidence into accurate numerical predictions. To address these challenges, we propose CastFlow, a role-specialized agentic forecasting framework that bridges general-purpose tool reasoning and specialized temporal forecasting. Specifically, CastFlow introduces an Experience Memory that distills successful tool-use trajectories from training data, guiding a frozen LLM to orchestrate a multi-view toolkit for evidence acquisition. A lightweight specialized LLM then performs evidence-grounded temporal reasoning to refine an ensemble forecasting prior. To enhance this capability, we develop a two-stage training strategy combining supervised fine-tuning and reinforcement learning with verifiable rewards, where final forecasting outcomes of the overall workflow guide the optimization of the specialized model. Extensive experiments on ten real-world benchmarks demonstrate that CastFlow outperforms competitive baselines on nine datasets. Our findings highlight the importance of bridging autonomous evidence acquisition and specialized temporal reasoning for accurate and adaptive agentic forecasting. The source code is available at https://github.com/Forever-Pan/CastFlow.

Bokai Pan, Mingyue Cheng, Xiaoyu Tao, Zhiding Liu, Shuo Yu, Yuchong Wu, Qi Liu, Defu Lian, and Enhong Chen
State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China, Hefei 230026, China.
E-mail: mycheng@ustc.edu.cn (Mingyue Cheng, corresponding author for this manuscript).

Keywords Time series forecasting · Large language models · Agentic forecasting · Evidence acquisition · Forecast refinement · Reinforcement learning with verifiable rewards

1 Introduction

Time series forecasting (TSF) supports data-driven decisions in energy and water systems, with applications including renewable energy generation forecasting [1] and streamflow prediction [2]. Accurate forecasting is challenging because temporal dynamics evolve, dependencies vary, and historical patterns can change abruptly [3, 4, 5]. Conventional forecasting models primarily learn mappings from past observations to future values [6, 7]. The emergence of large language model (LLM) agents creates an opportunity to augment these mappings with active investigation: agents can inspect temporal patterns, invoke analytical tools, and incorporate diagnostic evidence into forecasting decisions [8, 9]. Viewed as an analytical workflow, agentic forecasting must coordinate historical data, tool execution, and reusable experience, connecting predictive modeling with data access and model management [10, 11].

Recent studies have extended LLM-based forecasting through inference-time temporal reasoning, reinforcement fine-tuning, and experience-conditioned reasoning [12, 13, 14]. Tool-assisted frameworks further support diagnostic analysis, reflective revision, and learned sequential tool-use policies [15, 16, 17]. These advances enable forecasting workflows to investigate and revise predictions, but tool access and multi-step reasoning alone do not determine which evidence is useful or how it should alter future numerical values. Our focus is the coordination between experience-guided evidence

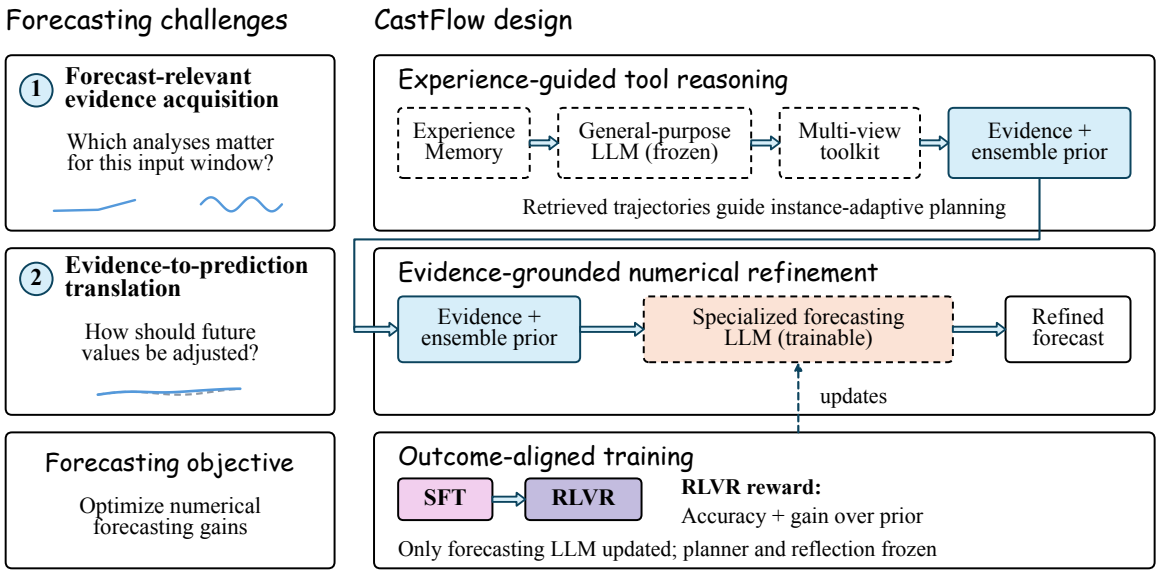


Fig. 1 Conceptual motivation for CastFlow. Experience-guided tool reasoning addresses forecast-relevant evidence acquisition, while a specialized LLM translates diagnostic evidence and an ensemble prior into refined numerical forecasts. SFT followed by RLVR updates only the forecasting LLM; the RLVR reward combines absolute prediction accuracy with improvement over the prior, while planning and reflection remain frozen

acquisition and a selectively trained numerical forecaster: retrieved procedures guide what to inspect, while forecasting supervision teaches how to use the resulting context. This motivates our central question: *How can LLM agents acquire forecast-relevant evidence and translate it into accurate time series predictions?*

Two linked challenges make this translation difficult. The first is *forecast-relevant evidence acquisition*: an agent must identify which observations and analyses matter for the temporal dynamics of the current instance. For example, an electricity-price sequence undergoing a recent trend shift may call for different diagnostics from one dominated by recurring seasonal patterns. Applying the same tool plan to every instance can introduce irrelevant evidence and unnecessary computation, whereas planning anew without task experience may overlook useful analyses. The agent therefore needs guidance for adapting its investigation to the observed context, using only information available at the forecast origin. The second challenge is *evidence-to-prediction translation*: heterogeneous diagnostic signals must inform precise numerical values across the forecasting horizon. Recognizing an upward trend or a possible turning point does not specify how much each future value should change. General-purpose reasoning can organize such information, but learning how it should inform numerical adjustments calls for adaptation to forecasting-specific objectives.

Our key insight is that evidence acquisition and numerical refinement can use different forms of adaptation. Retrieved procedural experience guides a general-purpose LLM in adapting tool plans to the current instance, while forecasting-specific supervision adapts a specialized model's parameters to numerical prediction. We therefore propose *CastFlow*, a role-specialized agentic framework that bridges evidence acquisition and accurate prediction through complementary model responsibilities. A frozen general-purpose LLM plans tool use and checks forecasts for consistency with the acquired evidence, providing feedback for bounded revision. A lightweight specialized LLM uses the diagnostic context and ensemble forecasting prior to generate refined numerical forecasts. Figure 1 illustrates how these complementary roles address the two challenges, supported by outcome-aligned training.

To address evidence acquisition, CastFlow builds an *Experience Memory* from training-time tool-use trajectories. For each training instance, candidate exploration paths are evaluated by horizon-wide forecasting error; the path with the lowest MSE among those passing format and logic checks is retained as structured experience containing the tool schedule, diagnostic observations, and final forecasting response. The memory is fixed before test inference, and the observed input window retrieves relevant entries to guide the frozen planner. Retrieved procedures inform the choice of optional diagnostics; the ensemble prior and exogenous analysis are always included. The multi-view toolkit supplies an ensemble prior and complementary evidence

on statistical and spectral structure, local temporal dynamics, and cross-variable dependencies. Tool execution grounds this evidence in information available at the forecast origin, including previously issued auxiliary forecasts where applicable.

To address evidence-to-prediction translation, the specialized LLM conditions its forecasting response on the acquired diagnostics and ensemble prior. The prior supplies a quantitative starting point, and diagnostic evidence informs adjustments to its temporal trajectory; the model generates the complete refined forecast conditioned on this context. We develop a two-stage training strategy combining supervised fine-tuning (SFT) with reinforcement learning using verifiable rewards (RLVR). SFT teaches the forecasting response protocol and the use of diagnostic context in temporal reasoning and numerical generation. RLVR then optimizes the specialized model using verifiable rewards computed from its final forecasting response under workflow-generated context, combining absolute prediction accuracy with improvement over the ensemble prior. Only the specialized forecasting model is optimized; the general-purpose planning and reflection model remains frozen throughout both stages.

We evaluate CastFlow on ten real-world benchmarks covering multiple application domains and short- and long-term forecasting settings. Among the evaluated methods, CastFlow achieves the lowest mean squared error (MSE) and mean absolute error (MAE) on nine datasets. Component and training ablations examine forecasting performance, while separate execution studies evaluate diagnostic feature reuse and controlled interruption recovery. These findings support combining experience-guided evidence acquisition with specialized, outcome-aligned numerical refinement to improve forecasting accuracy.

Our main contributions are as follows:

- We propose CastFlow, a role-specialized agentic framework that couples experience-guided evidence acquisition with prior-conditioned numerical refinement, using a frozen planning and reflection model and a selectively trained forecasting LLM.
- We construct a retrievable memory by selecting valid training trajectories according to forecasting error, and use it to guide instance-adaptive tool planning. SFT initializes the forecasting model, and RLVR rewards absolute accuracy and improvement over the ensemble prior.
- CastFlow achieves the lowest MSE and MAE on nine of ten real-world benchmarks among the evaluated methods. Component and training ablations, memory analyses, and controlled execution studies characterize its forecasting and workflow behavior.

2 Related Work

2.1 Traditional Time Series Forecasting

Time series forecasting spans statistical, machine learning, deep learning, and foundation models [7,6,18]. Classical approaches such as ARIMA [19] and ETS [20] characterize trend, seasonality, and temporal dependence through explicit structural assumptions, offering interpretable and efficient forecasting baselines [21]. Machine learning methods extend these baselines using lagged observations and engineered features, including support vector regression [22], boosting [23,24], and feature-based strategies [25]. Deep architectures reduce manual feature engineering and capture temporal patterns through diverse inductive biases [26,27], including linear, convolutional [28], Transformer [29,30], and state-space models [31], with representative methods such as N-HiTS [32], ETSformer [33], iTransformer [34], and ConvTimeNet [35]. Chronos [36], TimesFM [37], and Sundial [38] extend forecasting through large-scale pretraining with promising zero-shot and cross-domain capabilities, while ZooCast [39] explores model-zoo selection for zero-shot prediction. These methods typically emphasize predictive modeling from historical observations, with limited support for test-time tool interaction, explicit evidence acquisition, and iterative forecast revision [9].

2.2 LLM-Based Time Series Forecasting

Research on LLM-based time series modeling spans multiple tasks and adaptation strategies [40,41]. In forecasting, prominent approaches include prompt reformulation, input reprogramming, and semantic alignment [42, 43]. PromptCast [44] reformulates numerical sequences as textual prompts, Time-LLM [45] reprograms frozen LLMs with patch-level temporal embeddings, and S^2IP-LLM [46] and TokenCast [47] explore semantic alignment and token-based modeling. Numerical tokenization and uncertainty calibration also affect pretrained LLM forecasts [48]. Alongside these adaptations, TimeReasoner [12] studies slow-thinking temporal reasoning, while TimeSeriesScientist [15] develops agentic time series analysis. AlphaCast [16] combines diagnostic context, clustered case-based model references, and reflective LLM forecast refinement. Time-R1 [13] uses reinforcement fine-tuning to strengthen multi-step temporal reasoning, while Cast-R1 [17] learns tool-augmented sequential policies through SFT and multi-turn RL. MemCast [14] organizes historical patterns, reasoning experience, and general laws into hierarchical memory to

condition forecasting; FLAIRR-TS [49] coordinates retrieval, forecasting, and prompt-refinement agents without weight updates. These studies emphasize reasoning, interaction, and experience reuse [9]. CastFlow shares prior-guided numerical refinement with AlphaCast, but separates adaptation mechanisms: procedural memory guides tool planning, planning and reflection remain frozen, and SFT and RLVR update only the specialized forecasting model. This differs from Cast-R1's learned sequential tool-use policy and FLAIRR-TS's test-time prompt optimization.

2.3 Tool Use, Memory, and Post-Training

Toolformer [50] shows how language models can learn to invoke external tools, ReAct [8] interleaves reasoning with actions, and DEPS [51] supports interactive planning in complex environments. DocETL [52] optimizes document pipelines through agent-guided rewriting and evaluation; LLM-PQA [53] connects natural-language prediction queries with data lakes and model zoos for model retrieval or on-demand training. For time series anomaly detection, AnomaMind [54] combines tool-augmented reasoning with a structured workflow. Self-Refine [55] studies iterative refinement through self-feedback, while Reflexion [56] uses verbal reinforcement and episodic memory for self-correction across reasoning trajectories. MemoTime [57] uses experience memory for temporal knowledge graph question answering, illustrating memory use beyond numerical forecasting. STaR [58] studies self-improvement through generated reasoning traces, while DeepSeek-R1 [59] and Qwen3 [60] develop reasoning capabilities through post-training that includes reinforcement learning. In adjacent temporal tasks, TimeOmni-1 [61] studies reward-driven time series reasoning, whereas AlphaAgentEvo [62] applies self-evolving agentic RL to financial alpha mining. CastFlow combines these mechanisms through complementary roles: a frozen LLM acquires diagnostic evidence, while a trainable forecasting model refines an ensemble forecast prior.

3 Preliminaries

3.1 Problem Definition

At forecast origin t, let $\mathbf{x} \in \mathbb{R}^{L \times C_x}$ denote the L-step history ending at t, comprising target observations and available covariates, and $\mathbf{y} \in \mathbb{R}^{H \times C_y}$ the next H target steps. Optional covariate forecasts $\mathbf{u}^{[t]} \in \mathbb{R}^{H \times C_u}$ cover the same horizon and were issued by t. Here, C_x, C_u, and C_y are the corresponding channel counts, with $C_y = 1$ in the evaluated tasks. A numerical output of the complete forecasting procedure F is

$$\hat{\mathbf{y}} = F(\mathbf{x}, \mathbf{u}^{[t]}) \in \mathbb{R}^{H \times C_y}, \tag{1}$$

with $\mathbf{u}^{[t]}$ omitted when unavailable. Forecast error is measured by MSE and MAE; model-specific training objectives are given in Section 4.4.

3.2 Information Availability

Let $a(v)$ denote the availability time of source value v. Admissible source information is

$$\mathcal{I}_t = \{v : a(v) \leq t\}. \tag{2}$$

Inputs must belong to $\mathcal{I}_t$; diagnostics are derived from these inputs and fixed offline resources. Supplied forecasts remain fixed within each request; later revisions and realized future covariates are excluded. At test time, model parameters and Experience Memory remain fixed, while request-local diagnostics and feedback may evolve. The current request's target sequence is used only for subsequent evaluation. Appendix D details offline target use and dataset provenance.

4 Methodology

CastFlow connects experience-guided evidence acquisition with evidence-grounded numerical forecasting. We first describe its role-specialized architecture, then explain how historical experience guides tool planning, how a specialized model refines an ensemble forecasting prior, and how forecasting outcomes supervise this model.

4.1 Framework Overview

As illustrated in Fig. 2, CastFlow comprises planning, action, forecasting, and reflection stages. A general-purpose LLM with frozen parameters Θ performs tool planning and evidence-consistency assessment. A lightweight forecasting LLM with trainable parameters θ generates numerical predictions from the resulting context. The action stage is a deterministic execution interface: it invokes the planned tools and returns their outputs without introducing another trainable reasoning model.

The framework separates offline preparation from online inference. Offline, candidate tool-use trajectories on training instances are evaluated against observed targets to construct an Experience Memory. Teacher-generated forecasting responses then support supervised fine-tuning (SFT), followed by reinforcement learning

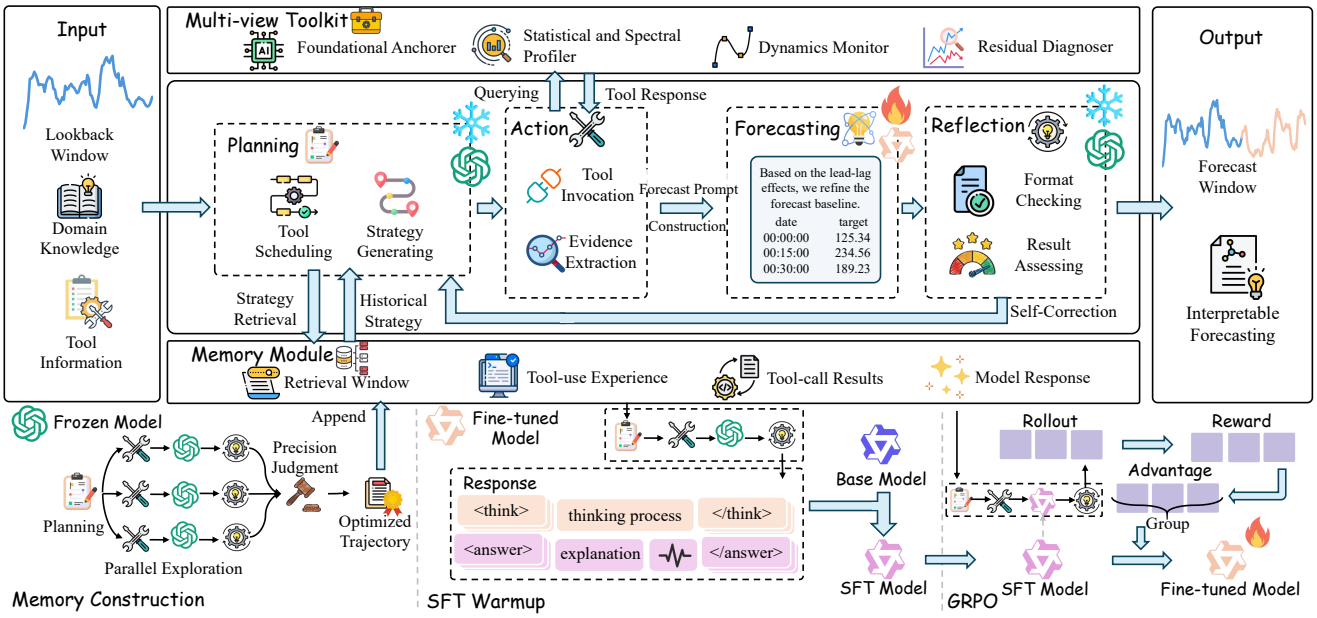


Fig. 2 Overview of CastFlow. Planning and reflection use a frozen general-purpose LLM, while the specialized forecasting LLM refines the ensemble prior using acquired evidence. The Memory Module denotes training-built Experience Memory and is read-only at test time; the residual diagnoser executes only during training. Online inputs include origin-issued horizon covariate forecasts where supplied. Reflection checks format and evidence consistency. Offline SFT and response-level GRPO update only the forecaster

with verifiable rewards (RLVR). Online, the framework retrieves historical experience, plans and executes tools using information available at the forecast origin, and supplies the resulting diagnostic evidence and ensemble prior to the specialized forecaster. Reflection checks the response and can initiate bounded revision. Ground-truth forecasting targets are used for offline selection and training rewards, not for online planning or verification.

Let $\mathcal{M}$ denote the offline Experience Memory and $\mathcal{M}_{\text{ret}}$ its retrieved entries. For an inference round j, the information flow is summarized by

$$\begin{aligned} \mathcal{M}_{\text{ret}} &= \text{Retrieve}(\mathbf{x}, \mathcal{M}), \\ A_j &= \text{Plan}_{\Theta}(\mathbf{x}, \mathcal{M}_{\text{ret}}, \mathcal{H}_j), \\ (\hat{\mathbf{y}}_{\text{base},j}, \mathcal{D}_j) &= \text{Execute}(A_j, \mathbf{x}, \mathbf{u}^{[t]}), \\ r_j &\sim \pi_\theta(\cdot \mid \mathcal{C}_j). \end{aligned} \tag{3}$$

Here, A_j is a tool schedule, $\mathcal{H}_j$ contains available workflow observations and feedback, and $\mathcal{C}_j$ is the forecasting context assembled from the lookback, origin-issued horizon covariate forecasts, ensemble prior, and diagnostic evidence. The response r_j contains the forecasting model's reasoning and numerical output. The initial history is empty; a failed verification supplies feedback for a subsequent round. The roles of Θ and θ remain unchanged across these rounds. Sections 4.2–4.3 describe this inference process, and Section 4.4 explains selective training.

4.2 Experience-Guided Evidence Acquisition

Evidence acquisition determines which observations accompany the numerical forecasting prior. CastFlow first retrieves structured experiences from training instances, uses them to guide tool planning, and executes the resulting schedule to obtain complementary diagnostic views. For a workflow round j, this stage produces the ensemble prior $\hat{\mathbf{y}}_{\text{base}}$ and diagnostic evidence $\mathcal{D}_j$ for the specialized forecaster.

For each training instance, CastFlow expands an initial plan into K_{exp} parallel exploration paths. Each path specifies a tool schedule, collects its outputs, and generates a candidate forecast. When at least one candidate passes both format and logic validation, we retain the valid path with the lowest horizon-wide MSE:

$$\tau^* = \arg\min_{\tau \in \mathcal{T}_{\text{valid}}} \frac{1}{HC_y} \sum_{h=1}^{H} \left\| \hat{\mathbf{y}}_{t+h}^{(\tau)} - \mathbf{y}_{t+h} \right\|_2^2, \tag{4}$$

where H is the forecasting horizon, C_y is the number of target channels, and $\mathcal{T}_{\text{valid}}$ contains the candidates accepted by both checks. Forecasting targets are used here to compare training paths. An archived entry is a structured tuple $e = \langle \mathbf{x}, A^*, O^*, r^* \rangle$, containing the lookback window, selected tool schedule, corresponding tool outputs, and the final forecasting response r^* of the selected trajectory τ^*. Distillation thus means selecting and retaining structured experiences from the explored

paths. Here, a successful trajectory means the valid, lowest-error candidate among the explored paths; selection does not impose an additional requirement that it outperform the ensemble prior.

The Experience Memory used in the main evaluation is constructed from training instances and their training targets, and is fixed before test inference. Test requests only retrieve existing records: their responses and targets are not written back to the memory, and test errors do not select or re-rank stored experiences. Append and merge concern the organization of training experience before this fixed-memory evaluation, rather than label-driven adaptation during testing. This separation also distinguishes persistent historical experience from the request-local observations and reflection feedback accumulated in $\mathcal{H}_j$.

Entries are indexed by vector similarity. For a new lookback window $\mathbf{x}$, retrieval returns up to K highest-similarity entries satisfying $\mathrm{sim}(\mathbf{x}, \mathbf{x}_e) \geq \eta$, where η is the similarity threshold. These entries provide examples of tool use and its associated evidence; their training outcomes support historical selection without requiring the current forecasting target. Memory influences prediction through two routes: retrieved schedules guide the planner's choice of diagnostics, and retrieved procedural guidance also conditions the forecasting response. Stored observations and responses describe historical training instances; evidence about the current window is produced by its own tool executions. Experience Memory provides procedural context for planning and forecasting, whereas the anchorer's historical library supplies model-selection and weighting information for the ensemble prior in Section 4.3.

Experience-Guided Tool Planning The frozen planner maps the input window and retrieved experiences $\mathcal{M}_{\mathrm{ret}}$ to a structured tool schedule A, following $P(A \mid \mathbf{x}, \mathcal{M}_{\mathrm{ret}}; \Theta)$. The schedule always includes the *model auxiliary tool* and *exogenous analysis tool*; other diagnostics are selected for the current sequence. Retrieved schedules provide procedural context for choosing these optional tools, while their observations indicate what evidence supported the associated forecasts.

On the first attempt, planning uses the current input and retrieved experiences. If a candidate forecast fails format or logic validation, reflection supplies natural-language feedback for a subsequent planning attempt within the bounded refinement loop. The planner selects evidence sources, and the action module executes the selected tools deterministically to return the ensemble prior and diagnostic evidence. Numerical prediction and forecast validation are described in the following subsection.

Multi-View Evidence Acquisition The toolkit comprises eleven tools in four functional categories; Appendix A provides the per-tool inventory and execution scopes (Table 7). The foundational anchorer returns an ensemble prior from forecasting models selected using historical performance on related windows; its retrieval and aggregation are detailed in Section 4.3. The remaining categories characterize the observed sequence from complementary perspectives.

The main inference protocol supplies both the observed lookback and covariate forecasts for the prediction horizon that were issued no later than the forecast origin t. We denote the latter by $\mathbf{u}^{[t]}$. These supplied forecasts are held fixed for the request, including its reflection retries; later-issued revisions and realized future observations are excluded. The execution interface makes this information available to relevant tools, while each tool consumes the inputs supported by its function. In particular, lookback diagnostics do not require future covariate forecasts. Appendix D separates the input protocol, per-value admissibility, and dataset provenance.

The statistical and spectral profiler summarizes numerical scale and dispersion, including robust deviation measures, and uses spectral entropy to describe how power is distributed across frequencies. Its data-quality outputs quantify missingness and provide range bounds for interpreting candidate forecasts. The dynamics monitor examines trends, temporal differences, and lead–lag or cross-channel associations. Its event summary expresses dominant patterns as rise, fall, flat, or oscillation, providing qualitative context alongside numerical features.

The residual diagnoser fits an autoregressive proxy to the observed lookback and summarizes its residual mean and first-order autocorrelation. It runs only during training and is bypassed at inference. Its lookback-fitted residuals do not require future targets, and any such outputs retained in memory describe their historical training window. Standard diagnostic definitions are collected in Appendix A.

4.3 Evidence-Grounded Forecast Refinement

Tool reasoning contributes to forecasting through two complementary inputs: a numerical forecasting prior and diagnostics of the observed temporal structure. The specialized model uses these inputs to generate a refined forecast, while the frozen reflection model checks response validity and consistency with the available evidence.

Ensemble Forecasting Prior The foundational anchorer provides a numerical starting point through the mandatory model auxiliary tool. Its historical case library is constructed offline by partitioning past time series into sliding windows and applying K-medoids clustering [63]. Each cluster stores a medoid and historical model-performance information for a pool spanning statistical methods, deep learning forecasters, and pretrained time series foundation models. This library serves model selection and weighting; it is distinct from the Experience Memory of procedural tool-use trajectories. Clustered case-based model references and LLM forecast refinement are also used by AlphaCast [16]; CastFlow supplies the ensemble as a mandatory tool output and trains the specialized forecasting response through SFT and RLVR.

For an input $\mathbf{x}$, the anchorer retrieves the closest cluster $\mathcal{C}^*$ using a distance that combines dynamic time warping [64], Euclidean distance, and cosine similarity on z-score-normalized representations. Let $\mathcal{I}_{\mathcal{C}^*}$ index the associated forecasting models. The ensemble prior is

$$\begin{aligned} w_k &= \frac{\exp(-\mathcal{L}_k)}{\sum_{q\in\mathcal{I}_{\mathcal{C}^*}}\exp(-\mathcal{L}_q)}, \\ \hat{\mathbf{y}}_{\text{base}} &= \sum_{k\in\mathcal{I}_{\mathcal{C}^*}} w_k f_k(\mathbf{x}), \end{aligned} \tag{5}$$

where f_k is a forecasting expert and $\mathcal{L}_k$ is its historical validation loss used to derive weights for the retrieved cluster. These historical quantities are separate from the unknown targets of the current forecasting request. The ensemble supplies an initial estimate of future values; the diagnostic tools supply context for interpreting and revising that estimate.

The expert pool and its historical performance records are prepared offline. This pool is a separate configuration from the competing baselines; Appendix D distinguishes expert weighting, memory retrieval, and response-generation settings.

Specialized Temporal Reasoning For each round, the action stage constructs the core forecasting context

$$\mathcal{C}_j^{\text{core}} = \big(\mathbf{x}, \mathbf{u}^{[t]}, \hat{\mathbf{y}}_{\text{base},j}, \mathcal{D}_j\big). \tag{6}$$

The historical observations describe the measured sequence and historical covariates; $\mathbf{u}^{[t]}$ supplies ex ante auxiliary forecasts aligned with the prediction horizon. Diagnostic outputs characterize distributional and temporal patterns, and the ensemble prior supplies a numerical reference. The full context $\mathcal{C}_j$ comprises $\mathcal{C}_j^{\text{core}}$ together with retrieved procedural guidance and available revision feedback. The forecasting prompt can therefore use horizon-specific auxiliary forecasts without receiving the unknown future target or realized future covariate observations.

The specialized LLM generates a serialized response r_j, intended to contain a reasoning trace z_j and a numerical forecast:

$$\begin{aligned} r_j &\sim \pi_\theta(\cdot \mid \mathcal{C}_j), \\ \hat{\mathbf{y}}_j &= \text{Parse}(r_j). \end{aligned} \tag{7}$$

The parser returns a finite numerical sequence of the required shape for a structurally valid response and a failure symbol $\perp$ otherwise. The prior supplies a value at each forecast position, while trends, changes, and lagged associations provide context for the direction and timing of possible revisions. Origin-issued covariate forecasts supply conditions specific to the future interval. SFT examples and the accuracy-based rewards in Section 4.4 train the policy to map this combined context to a complete numerical forecast. The schematic in Fig. 7 illustrates how retrieved experience, diagnostic evidence, and the ensemble prior connect to the specialized forecaster. These inputs condition learned numerical generation; they do not specify deterministic correction coefficients or certify the generated reasoning trace.

Refinement can be analyzed by defining $e_{\text{base}} = \mathbf{y} - \hat{\mathbf{y}}_{\text{base}}$ and $\Delta_\theta = \hat{\mathbf{y}} - \hat{\mathbf{y}}_{\text{base}}$. Then

$$\begin{aligned} &\|\hat{\mathbf{y}} - \mathbf{y}\|_F^2 - \|\hat{\mathbf{y}}_{\text{base}} - \mathbf{y}\|_F^2 \\ &\quad = \|\Delta_\theta\|_F^2 - 2\langle e_{\text{base}}, \Delta_\theta\rangle_F. \end{aligned} \tag{8}$$

The Frobenius norm and inner product sum over the forecast horizon and target channels. A correction improves squared error when its alignment with the baseline residual outweighs its squared magnitude. Here, Δ_θ is the difference between two forecasts, not a separate residual-prediction head. The identity motivates evaluating improvement against the prior during training; it does not guarantee that every generated correction improves forecasting accuracy.

Forecast Verification and Bounded Revision After generation, the workflow combines deterministic format checks with a frozen-model assessment of evidence consistency:

$$v_j = \begin{cases} 0, & \mathbb{I}_{\text{format}}(r_j) = 0, \\ \mathbb{I}_{\text{logic}}(\hat{\mathbf{y}}_j, \mathcal{D}_j), & \mathbb{I}_{\text{format}}(r_j) = 1. \end{cases} \tag{9}$$

The format indicator checks whether the response can be parsed into finite numerical values with the required forecast length. For a structurally valid response, the logic indicator records the reflection model's assessment of its consistency with the diagnostic signals. This assessment has no access to the request's unknown forecasting targets and is not a measurement of true predictive error.

Algorithm 1 Bounded inference and fallback

Require: Lookback $\mathbf{x}$, origin-issued covariate forecasts $\mathbf{u}^{[t]}$, Memory $\mathcal{M}$, retry budget $C_{\max}$
Ensure: Numerical forecast $\hat{\mathbf{y}}_{\text{out}}$ or a failure status

$\mathcal{M}_{\text{ret}} \leftarrow \text{Retrieve}(\mathbf{x}, \mathcal{M})$
$\mathcal{H} \leftarrow \emptyset$; $\mathcal{R}_{\text{valid}} \leftarrow \emptyset$
for $j = 0, \ldots, C_{\max}$ **do**
  $A \leftarrow \text{Plan}_{\Theta}(\mathbf{x}, \mathcal{M}_{\text{ret}}, \mathcal{H})$
  $(\hat{\mathbf{y}}_{\text{base}}, \mathcal{D}) \leftarrow \text{Execute}(A, \mathbf{x}, \mathbf{u}^{[t]})$
  Assemble $\mathcal{C}$ from inputs, prior, evidence, and feedback
  Generate $r \sim \pi_\theta(\cdot \mid \mathcal{C})$
  if $\mathbb{I}_{\text{format}}(r) = 1$ **then**
    Add r to $\mathcal{R}_{\text{valid}}$
  end if
  $v \leftarrow \text{Verify}_{\Theta}(r, \mathcal{D})$
  if $v = 1$ **then**
    return Parse(r) ▷ Accepted
  end if
  Update $\mathcal{H}$ with observations, r, and feedback
end for
if $\mathcal{R}_{\text{valid}} \neq \emptyset$ **then**
  Choose $r_{\text{fb}} \in \mathcal{R}_{\text{valid}}$
  return Parse(r_{fb}) ▷ Response fallback
else if the ensemble prior is usable **then**
  return $\hat{\mathbf{y}}_{\text{base}}$ ▷ Prior fallback
else
  return missing or abnormal output status
end if

If $v_j = 0$, reflection supplies feedback for another planning attempt, which may revise the diagnostics or the forecasting prompt. The workflow permits one initial attempt and at most $C_{\max}$ retries. When the budget is exhausted without acceptance, the main evaluation first uses a structurally valid response from the request history, then falls back to the ensemble prior if no such response is available. Only when neither source provides a usable forecast is the output treated as missing or abnormal. This fallback order uses available request outputs without consulting forecasting targets. A fallback prediction is distinct from a response accepted by reflection. Algorithm 1 summarizes the inference and fallback sequence.

4.4 Outcome-Aligned Training

Forecasting outcomes supervise three distinct offline operations: selecting experiences for memory, selecting teacher responses for SFT, and scoring generated responses for RLVR. The first two retain useful historical procedures and examples; the third optimizes the specialized forecasting policy. Throughout SFT and RLVR, the planner and reflection model remain frozen. Their workflow supplies the conditioning context, while gradient updates apply exclusively to the forecasting model.

Supervised Fine-Tuning After constructing the Experience Memory in Section 4.2, we incorporate it into the workflow to obtain teacher supervision. For each training instance, a powerful teacher model generates candidate forecasting responses conditioned on retrieved procedural guidance, tool-derived diagnostics, and the ensemble prior. We select a preferred valid response using forecasting error over the target horizon, and extract the forecasting context–response pair $(\mathcal{C}, r^*)$ from the resulting teacher trajectory. The selected response contains a reasoning trace and numerical answer. Upstream plans and tool outputs form the conditioning context rather than additional trainable policies.

Writing the selected response as tokens $(w_1, \ldots, w_T)$, the forecasting model minimizes

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(\mathcal{C}, r^*)} \left[\sum_{i=1}^{T} \log \pi_\theta(w_i \mid w_{<i}, \mathcal{C}) \right]. \quad (10)$$

This stage teaches the local model to follow the forecasting response protocol and condition numerical generation on the workflow evidence. Forecasting targets are used to select teacher examples during offline preparation; they are not included as future observations in the inference context.

Verifiable Forecasting Rewards For a training instance with target $\mathbf{y}$, let r be a sampled forecasting response and $\hat{\mathbf{y}}(r)$ its final numerical prediction when accepted by the numerical reward parser. Each response is scored directly from this prediction; a separate reflection and retry loop is not executed for each GRPO candidate. Thus, final forecasting outcome refers here to the numerical answer in the response conditioned on workflow evidence. We evaluate its horizon-wide MSE and that of the ensemble prior supplied in the same context:

$$\begin{aligned} \mathcal{L}_{\text{agent}} &= \frac{1}{HC_y} \sum_{h=1}^{H} \|\hat{\mathbf{y}}_{t+h}(r) - \mathbf{y}_{t+h}\|_2^2, \\ \mathcal{L}_{\text{base}} &= \frac{1}{HC_y} \sum_{h=1}^{H} \|\hat{\mathbf{y}}_{\text{base},t+h} - \mathbf{y}_{t+h}\|_2^2. \end{aligned} \quad (11)$$

Both errors refer to the same instance and horizon. The context and its ensemble prior remain fixed within a GRPO group, so each candidate is compared with the same numerical starting point. The reward combines numerical validity, absolute accuracy, and improvement over that prior:

$$\begin{aligned} R(r) &= \begin{cases} -\mathcal{P}_{\text{violation}}(r), & r \notin \mathcal{V}, \\ R_{\text{abs}}(\mathcal{L}_{\text{agent}}) + R_{\text{rel}}(r), & r \in \mathcal{V}, \end{cases} \\ R_{\text{rel}}(r) &= \text{Clip}\left(\lambda \frac{\mathcal{L}_{\text{base}} - \mathcal{L}_{\text{agent}}}{\nu_d}, -\delta, \delta \right). \end{aligned} \quad (12)$$

Here, $\mathcal{V}$ contains responses accepted by the numerical reward parser with the required forecast length and a non-NaN candidate MSE, and $\mathcal{P}_{\text{violation}}(r) > 0$ penalizes unusable answers or length mismatches. For dataset d, the absolute term is

$$R_{\text{abs}}(\epsilon) = \begin{cases} 1 - 0.9 \sin\left(\dfrac{\pi\epsilon}{2\gamma_d}\right), & 0 \le \epsilon \le \gamma_d, \\ 0.3 \exp\left(-\dfrac{\ln(3)\epsilon}{\gamma_d}\right), & \epsilon > \gamma_d. \end{cases} \tag{13}$$

The branches meet at $R_{\text{abs}}(\gamma_d) = 0.1$. The relative term uses $\lambda = 5$ and $\delta = 0.5$. Its denominator ν_d is a configured dataset-specific MSE scale, so it measures a scaled error difference rather than a percentage change relative to baseline MSE. Appendix D.6 gives all dataset scales and output penalties.

The reward is computed directly from each response using offline targets. Deterministic parsing and error computation provide the verifiable training signal. The reward parser determines numerical validity; the reflection model's semantic decision is not part of $\mathcal{V}$. Accordingly, a candidate accepted by the reward parser can receive a numerical reward independently of whether reflection would accept it during inference. Reflection and fallback govern deployed inference as described in Section 4.3; they are not additional GRPO scoring stages. The reward encourages accurate refinement without directly verifying the faithfulness of a reasoning trace.

Selective Policy Optimization We use group relative policy optimization (GRPO) [65] to optimize the forecasting response policy without a separate value network. For each workflow-generated forecasting prompt $\mathcal{C}$, the policy samples a group of G responses while holding the observations, diagnostic outputs, procedural guidance, and ensemble prior fixed. Their rewards are compared within the group; for nonzero reward variance, the normalized advantage is

$$A_i = \frac{R_i - \mu_R}{\sigma_R}, \tag{14}$$

where μ_R and σ_R are the group reward mean and standard deviation. The outcome-aligned objective can be written as

$$J(\theta) = \mathbb{E}_{\mathcal{C} \sim d_{\mathcal{W}},\, r \sim \pi_\theta(\cdot|\mathcal{C})}[R(r)]. \tag{15}$$

Here, $d_{\mathcal{W}}$ denotes the distribution of forecasting contexts supplied by the workflow with frozen planning and reflection, and $R(r)$ scores the numerical answer generated under a given context. The objective expresses candidate-level optimization of evidence-conditioned forecasts. The same scalar outcome reward supplies the advantage for the generated response tokens, including its reasoning and answer; upstream plans and tool outputs remain conditioning inputs. Planning and reflection receive no policy updates. Appendix C describes the surrounding decision process, while the trainable objective above concerns the specialized forecasting response.

5 Experiments

5.1 Experimental Settings

Datasets Table 1 summarizes the ten benchmarks, target variables, and forecasting configurations. For short-term forecasting, we use the BE, DE, FR, NP, and PJM markets from the EPF benchmark [66], distributed in the TimeXer collection [67]. Each contains a target electricity price series and two market-specific exogenous forecast series. In the long-term setting, the widely used ETTh and ETTm datasets [68] are utilized to monitor transformer temperature and load variations under different time granularities. To capture the dynamics of renewable energy generation, we adopt the wind power (WP) and solar power (SP) datasets [1], which align real power generation records with multi-dimensional meteorological conditions. Additionally, the MOPEX dataset [2] is included for streamflow forecasting, featuring streamflow series supported by climatic factors. For rigorous evaluation, all multivariate time series datasets are strictly split chronologically into training, validation, and testing sets with a consistent ratio of 7:1:2.

Baselines To provide a comprehensive evaluation, we compare CastFlow against 21 representative baselines spanning five distinct methodological paradigms, covering both conventional and emerging approaches, from classical forecasting to recent reasoning-oriented frameworks under the dataset and metric protocol described below: (1) statistical models: Prophet [69] and ARIMA [19]; (2) machine learning models: XGBoost [23] and LightGBM [24]; (3) deep learning forecasters: Autoformer [70], DLinear [71], PatchTST [72], iTransformer [34], TimeXer [67], and ConvTimeNet [35]; (4) foundation models: Chronos [36], TimesFM [37], and Sundial [38]; (5) large language model (LLM)-based and agentic frameworks: Time-LLM [45], PromptCast [44], TokenCast [47], S^2IP-LLM [46], TimeReasoner [12], Time-R1 [13], TimeSeriesScientist [15], and AlphaCast [16]. These baselines provide a solid basis for evaluating our method across diverse benchmarks.

Implementation Details We utilize Grok 4 [73] as the frozen backbone model for general-purpose reasoning during both training and testing phases, which also

Table 1 Summary of the benchmark datasets for time series forecasting. "Dim" denotes the total number of recorded columns, including the timestamp column, while the feature list reports only the target variable and covariates. Shared feature lists apply to both ETT rows and both renewable-power rows. All datasets use a 7:1:2 training/validation/test split.

Dataset	Domain	Frequency	Dim	L	H	Stride	Features (Target & Covariates)
BE	Electricity Market	1h	4	168	24	48	Electricity price, Generation forecast, System load forecast
DE	Electricity Market	1h	4	168	24	48	Electricity price, Wind power forecast, Amprion zonal load forecast
NP	Electricity Market	1h	4	168	24	48	Electricity price, Grid load forecast, Wind power forecast
FR	Electricity Market	1h	4	168	24	48	Electricity price, Generation forecast, System load forecast
PJM	Electricity Market	1h	4	168	24	48	Electricity price, System load forecast, Zonal COMED load forecast
ETTh	Power Grid	1h	8	96	96	48	Oil temperature, High useful load, High useless load, Middle useful load, Middle useless load, Low useful load, Low useless load
ETTm	Power Grid	15min	8	96	96	96	
WP	Renewable Energy Generation	15min	8	96	96	96	Real power, Direct radiation, Wind direction 80m, Wind speed 80m, Temperature 2m, Relative humidity 2m, Precipitation
SP	Renewable Energy Generation	15min	8	96	96	96	
MOPEX	Streamflow	1day	6	96	96	48	Streamflow discharge, Mean areal precipitation, Climatic potential evaporation, Maximum air temperature, Minimum air temperature

facilitates experience generation as a teacher model during memory construction. Meanwhile, Qwen3-4B [60] is employed specifically as the trainable local LLM for specialized numerical forecasting, configured with a max completion length of 5,000 tokens. Local model experiments use two NVIDIA A800 GPUs; Grok 4 provides the frozen general-purpose reasoning and teacher responses. We implement the training pipeline using the transformers Trainer [74] for the supervised stage and the Agent Lightning framework [75] for the reinforcement learning with verifiable rewards (RLVR) stage. The process consists of two phases: (1) supervised fine-tuning (SFT) with a learning rate of 5×10^{-5} and batch size of 8, running for 1 epoch in cross-domain joint training; and (2) RLVR using group relative policy optimization (GRPO) [65] with a group size of $G = 8$, temperature of 1.0, and learning rate of 2×10^{-6}. The KL penalty coefficient is set to $\beta = 0.0$. The RLVR stage spans 3 epochs in cross-domain joint training.

For evaluation, we set the lookback window $L = 168$ and horizon $H = 24$ for short-term tasks, while long-term tasks use $L = 96$ and $H = 96$. All methods are evaluated using the same chronological splits, target variables, and forecasting horizons. Methods that support exogenous inputs receive covariates through their supported interfaces; target-only baselines use only the target series. CastFlow also uses the origin-issued horizon covariate forecasts described in Section 4.2. We follow official implementations or recommended configurations whenever available, including model-specific training schedules, early stopping, preprocessing, and prompt construction. Trainable baselines are fitted on the training split, while remaining hyperparameters and early-stopping criteria are selected on the validation split without test-set information. For prompt- or LLM-based baselines, contextual inputs are constructed according to their original protocols and supported covariate interfaces, restricted to information available at the forecast origin, without realized future observations or test labels. When model-specific preprocessing or normalization is applied, all reported metrics are computed after inverse transformation to the original target scale. CastFlow retains the original magnitude values in its forecasting context without explicit input-output normalization strategies such as RevIN [76].

Lookback lengths, forecasting horizons, and strides in Table 1 are measured in sampling steps, so their temporal spans depend on dataset-specific sampling

Table 2 MSE and MAE on ten forecasting benchmarks under cross-domain joint training. The two panels report short- and long-term tasks for the same models. Bold and underlined values indicate the best and second-best results, respectively.

Short-term forecasting

Model	BE		DE		NP		FR		PJM	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Prophet	992.90	16.94	320.63	13.25	57.79	5.00	1035.70	14.42	52.26	5.53
ARIMA	869.14	17.04	372.91	11.82	55.51	4.45	964.00	13.22	33.02	4.00
XGBoost	1344.39	16.44	227.01	10.21	28.31	3.24	1044.82	10.14	30.36	3.86
LightGBM	1168.62	15.71	231.44	9.57	30.92	3.38	933.84	9.96	29.63	3.88
Autoformer	891.02	16.62	331.37	13.33	46.11	4.86	934.55	13.37	77.59	6.43
DLinear	658.66	12.84	239.98	12.84	32.22	3.84	811.62	10.45	42.16	4.65
PatchTST	627.27	11.44	208.93	10.04	24.64	3.27	797.42	9.13	31.88	4.17
iTransformer	606.71	11.25	230.05	10.23	27.10	3.44	940.51	11.10	35.14	4.37
TimeXer	703.00	10.70	252.05	10.26	27.31	3.44	834.86	9.38	25.88	3.65
ConvTimeNet	621.73	10.98	231.26	10.80	28.13	3.62	882.85	8.97	28.71	3.79
Chronos	625.88	9.63	223.24	10.00	24.39	3.21	812.40	8.24	**25.70**	3.63
TimesFM	784.98	10.92	219.57	9.98	25.11	3.20	870.17	8.84	33.99	4.14
Sundial	651.50	11.00	264.72	10.71	28.68	3.36	942.46	8.54	30.72	3.99
Time-LLM	729.68	11.88	253.97	10.85	33.05	3.71	936.46	11.51	35.84	4.32
PromptCast	824.27	13.38	293.24	10.93	43.16	3.93	1105.16	14.42	47.62	5.58
TokenCast	654.38	11.48	232.21	10.84	31.49	3.59	963.09	11.82	32.37	3.92
S^2IP-LLM	701.24	11.89	313.82	11.15	29.67	3.42	872.55	10.93	35.67	4.16
TimeReasoner	691.28	11.62	272.19	10.77	31.27	3.62	899.47	11.01	34.38	4.22
Time-R1	635.23	10.23	210.28	9.89	26.28	3.35	842.19	10.92	27.35	3.80
TimeSeriesScientist	906.68	14.94	245.40	10.39	46.31	4.16	904.65	12.71	45.40	5.39
AlphaCast	653.98	10.71	248.98	10.23	24.16	3.19	804.89	9.10	26.53	**3.59**
CastFlow	**546.87**	**9.60**	**200.47**	**9.50**	**23.92**	**3.11**	**707.96**	**8.16**	27.45	3.74

Long-term forecasting

Model	ETTh		ETTm		WP		SP		MOPEX	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Prophet	46.70	4.59	21.18	3.11	8071.24	56.06	64.81	6.44	12.40	2.50
ARIMA	10.88	2.43	2.71	1.17	2106.64	28.77	80.48	4.60	9.19	1.85
XGBoost	13.37	2.96	3.41	1.37	2384.63	37.99	33.85	3.50	6.91	1.68
LightGBM	12.22	2.75	2.91	1.26	2290.27	34.81	29.50	2.80	9.63	1.84
Autoformer	11.60	2.64	4.23	1.57	2568.13	37.01	39.14	4.34	6.17	1.96
DLinear	8.51	2.24	2.63	1.14	1932.59	29.33	19.06	2.74	5.26	1.33
PatchTST	8.40	2.17	2.62	1.11	2270.10	32.47	19.44	2.78	5.36	1.44
iTransformer	8.31	2.14	3.14	1.28	2064.60	29.90	17.96	2.67	4.88	1.34
TimeXer	8.77	2.20	2.56	1.11	2170.25	31.92	20.12	2.92	4.83	1.28
ConvTimeNet	8.62	2.18	2.71	1.16	2081.37	28.52	23.22	3.10	5.07	1.29
Chronos	9.40	2.25	2.75	1.18	2269.43	29.63	79.28	4.57	5.29	1.31
TimesFM	9.20	2.26	2.78	1.15	2228.19	29.84	45.91	3.37	5.58	1.35
Sundial	9.44	2.32	3.21	1.24	2168.81	31.22	100.40	6.61	5.15	1.42
Time-LLM	8.04	2.08	2.57	1.13	2833.24	38.05	17.25	2.68	5.77	1.44
PromptCast	9.46	2.82	4.25	1.60	2626.89	36.20	18.21	2.84	6.70	1.59
TokenCast	8.28	2.17	2.50	1.10	2521.39	34.29	18.13	2.74	5.52	1.39
S^2IP-LLM	9.68	2.39	3.01	1.29	2497.38	33.13	18.46	2.96	6.29	1.58
TimeReasoner	9.14	2.18	4.04	1.57	2228.24	31.57	20.91	2.98	5.92	1.46
Time-R1	8.02	2.04	2.80	1.11	2054.88	30.13	17.42	2.67	5.83	1.35
TimeSeriesScientist	9.09	2.24	5.28	1.61	2551.19	33.59	25.82	3.14	5.60	1.42
AlphaCast	8.47	2.21	2.48	1.16	2131.02	30.22	18.89	2.80	5.19	1.43
CastFlow	**8.00**	**2.03**	**2.36**	**1.04**	**1719.99**	**27.40**	**16.90**	**2.50**	**3.60**	**1.19**

intervals, while strides determine the spacing between forecast origins. A 24-step horizon spans 24 hours on BE, DE, FR, NP, and PJM, whereas a 96-step horizon corresponds to 96 hours on ETTh, 24 hours on ETTm, WP, and SP, and 96 days on MOPEX.

Generation and ablation reporting. The sampling parameters above describe GRPO training. Greedy decoding in the recovery experiment is specific to that execution comparison. Appendix D.3 clarifies the scope of the reported settings.

Table 3 Component ablation under the evaluated configurations. "w/o Toolkit" removes the ensemble prior and diagnostics together with dependent tool planning; "w/o Memory" removes experience retrieval; "w/o Refl." removes self-correction. The complete configuration has the lowest reported errors in this table.

Settings		Short-term Forecasting					Long-term Forecasting				
Metrics	Variants	BE	DE	NP	FR	PJM	ETTh	ETTm	WP	SP	MOPEX
MSE	w/o Refl.	1027.76	394.83	38.64	801.89	45.06	33.84	20.51	5102.29	22.72	5.18
	w/o Toolkit	836.24	358.33	31.21	724.87	42.33	23.71	17.68	3517.03	21.46	4.54
	w/o Memory	551.54	204.54	24.67	712.30	28.30	8.45	2.41	1820.92	17.49	4.63
	Full Model	**546.87**	**200.47**	**23.92**	**707.96**	**27.45**	**8.00**	**2.36**	**1719.99**	**16.90**	**3.60**
MAE	w/o Refl.	15.35	12.44	4.07	10.55	5.47	5.37	4.27	43.48	3.85	1.79
	w/o Toolkit	13.53	11.50	3.62	9.46	4.77	3.55	2.86	37.52	2.96	1.36
	w/o Memory	9.62	9.77	3.14	8.22	3.79	2.18	1.08	28.69	2.55	1.38
	Full Model	**9.60**	**9.50**	**3.11**	**8.16**	**3.74**	**2.03**	**1.04**	**27.40**	**2.50**	**1.19**

Workflow execution support. We use request-local diagnostic feature reuse to avoid repeated primitive computations and a persistent execution journal to resume committed stages after interruption. Reuse requires matching inputs and configuration; a semantic revision produces a new forecasting stage. Appendix B gives the mechanisms and their scope, and Section 5.7 evaluates diagnostic computation and controlled interruption recovery.

5.2 Overall Forecasting Performance

Table 2 compares CastFlow with 21 baselines using MSE and MAE on ten datasets. CastFlow achieves the lowest values of both metrics on nine datasets, corresponding to 18 of the 20 dataset–metric comparisons. It leads on four of the five electricity-market datasets and all five datasets evaluated at the 96-step horizon. These results establish the effectiveness of the complete forecasting configuration across the evaluated tasks.

The gains vary by dataset and metric. On WP, MSE decreases from the strongest competing result of 1932.59 to 1719.99, a reduction of 11.00%; MAE decreases from 28.52 to 27.40, a reduction of 3.93%. On MOPEX, the corresponding reductions are 25.47% in MSE, from 4.83 to 3.60, and 7.03% in MAE, from 1.28 to 1.19. Each percentage uses the lowest competing value for the same dataset and metric as its reference. Improvements are smaller on several other comparisons, including BE MAE (9.63 to 9.60) and ETTh MSE (8.02 to 8.00). The table reports point estimates, so these small margins should not be interpreted as evidence of statistically significant superiority.

PJM is the exception: CastFlow obtains an MSE of 27.45 and an MAE of 3.74, compared with the best competing values of 25.70 and 3.59, respectively. The complete workflow therefore does not improve on the strongest baseline in every setting. The following analyses examine its components and training under the reported configurations; the aggregate comparison alone does not attribute the gains to a particular reasoning step or explain the PJM gap.

5.3 Experience-Guided Evidence Acquisition

We examine Experience Memory, trajectory organization and retrieval, and toolkit composition. The training-built library remains fixed during testing; test responses and targets are never written back. Diagnostics use origin-available inputs, including previously issued horizon covariate forecasts where supported (Appendix D).

Experience Memory Contribution The full configuration has lower MSE and MAE than the no-memory variant on all ten datasets (Table 3). Removing memory raises MOPEX MSE from 3.60 to 4.63 and MAE from 1.19 to 1.38, and ETTh MSE from 8.00 to 8.45. Relative to the no-memory variant, MOPEX improves by 22.25% MSE and 13.77% MAE. BE gains are smaller: MSE changes from 551.54 to 546.87 and MAE from 9.62 to 9.60, improvements of 0.85% and 0.21%. Memory utility is therefore task-dependent. Because guidance enters both planning and forecasting, this ablation measures their combined workflow contribution rather than tool selection alone.

Memory Organization and Retrieval Before the library is fixed, append retains trajectories separately while merge combines them with representative experiences. Append generally yields lower errors, particularly on BE and DE (Fig. 3(a)). Preserving individual tool schedules and diagnostic contexts offers a plausible explanation.

For retrieval scales $K \in \{1, 3, 5, 7\}$, BE and SP show comparatively stable MAE; other datasets improve

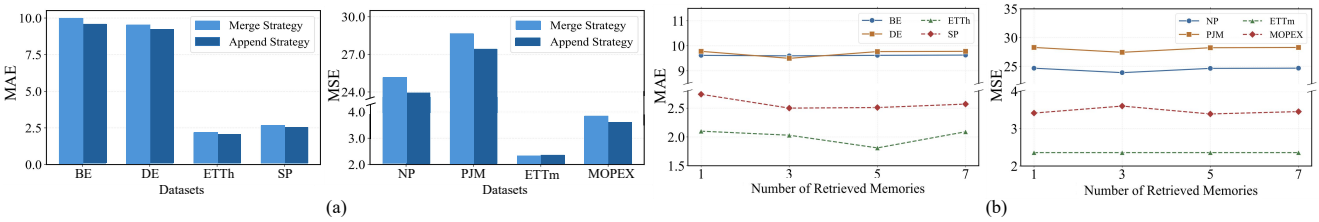


Fig. 3 Organization and retrieval of training experience. (a) Append and merge are compared before the Experience Memory is fixed for testing. (b) Forecasting error as the retrieval scale K varies

Table 4 Tool-category ablation. The anchorer supplies the ensemble prior, while the profiler and monitor supply diagnostics. The residual diagnoser (Diagn.) is available only during training. Best reported values are bolded.

Settings		Short-term Forecasting					Long-term Forecasting				
Metrics	Variants	BE	DE	NP	FR	PJM	ETTh	ETTm	WP	SP	MOPEX
MSE	w/o Anchorer	697.35	270.36	28.50	786.70	47.89	17.30	3.53	2718.42	20.17	5.67
	w/o Profiler	567.03	201.87	24.39	**675.67**	28.15	**7.88**	**2.33**	1821.36	17.42	4.69
	w/o Monitor	547.23	200.90	24.66	695.90	28.57	7.90	2.43	1890.65	19.38	4.52
	w/o Diagn.	553.86	201.67	24.48	710.00	28.18	8.08	2.38	1805.12	17.49	4.54
	Full Model	**546.87**	**200.47**	**23.92**	707.96	**27.45**	8.00	2.36	**1719.99**	**16.90**	**3.60**
MAE	w/o Anchorer	13.37	11.16	3.53	10.55	4.85	3.00	1.36	35.05	2.62	1.35
	w/o Profiler	9.71	9.68	3.13	**7.59**	3.78	2.06	1.05	28.40	2.51	1.37
	w/o Monitor	9.62	9.66	3.15	7.88	3.82	2.04	1.08	28.96	**2.45**	1.36
	w/o Diagn.	9.68	9.69	3.12	8.20	3.78	2.07	1.06	28.39	2.53	1.36
	Full Model	**9.60**	**9.50**	**3.11**	8.16	**3.74**	**2.03**	**1.04**	**27.40**	2.50	**1.19**

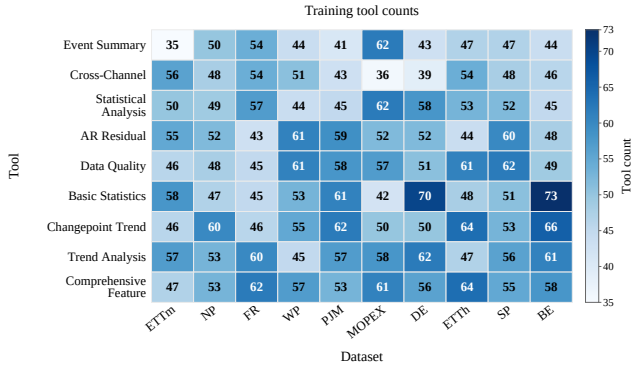


Fig. 4 Training-phase tool invocation counts across datasets. The two mandatory tools, model auxiliary and exogenous analysis, are omitted. The color scale covers all displayed counts (35–73). Counts describe schedules generated by the frozen planner

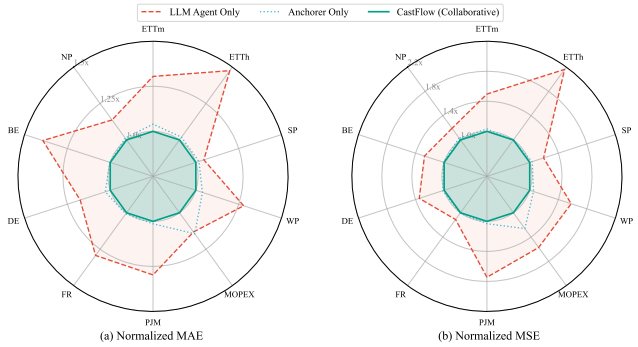


Fig. 5 Forecasting configurations relative to full CastFlow: (a) MAE and (b) MSE. LLM Agent Only uses Grok 4 with diagnostics but without an ensemble prior; Anchorer Only returns the prior; CastFlow adds the complete evidence-conditioned refinement workflow

before plateauing or deteriorating (Fig. 3(b)). More context is therefore not uniformly beneficial, and a small relevant collection can suffice in some settings.

Tool-Category Contributions Removing the anchorer produces the largest MSE increase among category removals on every dataset, including WP's increase from 1719.99 to 2718.42 (Table 4). This pattern confirms the baseline's central role in evidence-guided refinement. Other effects vary: removing the profiler improves FR MSE from 707.96 to 675.67 and MAE from 8.16 to 7.59, reductions of 4.56% and 6.99%, but raises MOPEX MSE from 3.60 to 4.69. The full configuration has the lowest MSE on seven datasets and MAE on eight. Adding diagnostics therefore does not uniformly improve refinement, motivating instance-relevant evidence selection. The residual diagnoser runs only during training; retained summaries describe training windows, not current test requests.

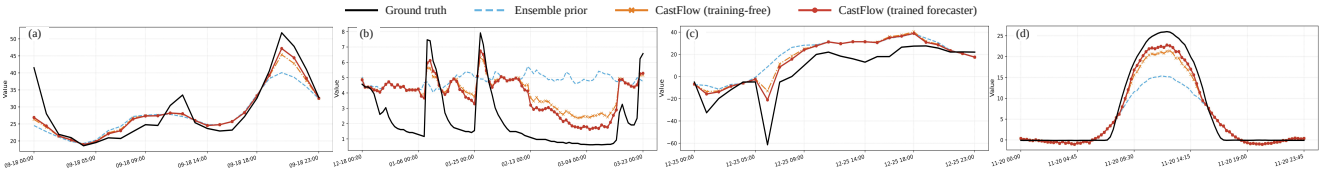


Fig. 6 Illustrative forecasting windows (a)–(d), comparing the ensemble prior, training-free CastFlow, and CastFlow with the trained forecasting model. Panel letters identify displayed examples; the original curves are retained

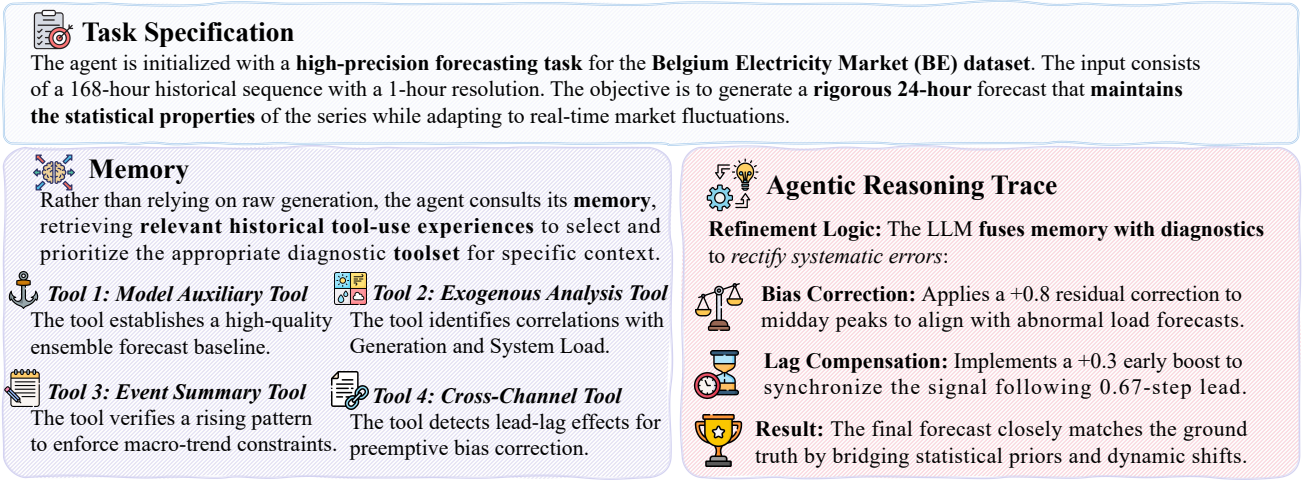


Fig. 7 Illustrative BE forecasting workflow with a 168-hour lookback and a 24-hour horizon. Retrieved experience guides tool planning; the ensemble prior and diagnostic evidence condition the forecasting response. The displayed trace describes peak and timing adjustments

Observed Tool Use Figure 4 shows nine optional tools, excluding mandatory model auxiliary and exogenous analysis. Statistical and trend tools are frequent, particularly on BE and DE; event summaries are less frequent. These training schedules describe the frozen planner, whose tool-selection policy is not directly updated by forecasting-model training.

5.4 Evidence-Grounded Forecast Refinement

Forecasting Configurations Figure 5 compares Grok 4 with diagnostics but no prior (LLM Agent Only), the ensemble prior (Anchorer Only), and full CastFlow. The first generally has larger normalized errors; the prior is competitive, and full refinement improves the overall profile. Both prior availability and forecasting-model adaptation differ, so this comparison does not isolate role specialization alone.

Toolkit Support and Response Verification Removing the toolkit raises WP MSE from 1719.99 to 3517.03 and ETTh MSE from 8.00 to 23.71 (Table 3); this removes the prior, diagnostics, and dependent planning together. Removing reflection yields WP MSE of 5102.29; this result does not isolate semantic reflection from structural checks or terminal output handling. Neither comparison isolates diagnostic reasoning or semantic reflection alone. Appendix D.5 specifies the fallback order used in the main evaluation when no response passes the verification checks within the retry budget.

Illustrative Forecast Refinement Figure 6 illustrates how refinement changes the ensemble prior in four windows. In panels (a) and (d), the trained forecaster's curve more closely follows the positive peak than the ensemble prior. The benefits have limits: the abrupt negative excursion in panel (c) remains poorly captured by both refinement configurations. These examples show changes in peak magnitude and local alignment together with remaining errors. They complement the forecasting tables as qualitative illustrations, without establishing population-level improvement in peak or lag correction.

The BE example in Fig. 7 connects refinement to diagnostic context for a 168-hour lookback and a 24-hour horizon. Retrieved experience guides the frozen planner's tool schedule. The model auxiliary tool establishes the ensemble prior; exogenous analysis, event summaries, and cross-channel associations provide context for the

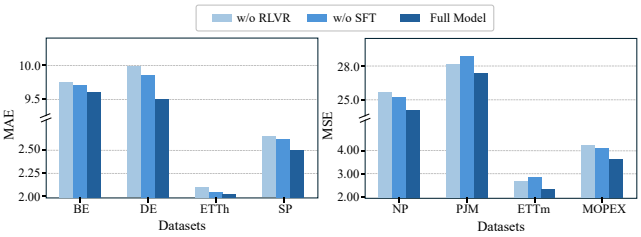


Fig. 8 Training-stage ablation for the specialized forecasting model. The full SFT followed by RLVR configuration is compared with variants omitting either stage using MAE and MSE; the planning and reflection model remains frozen

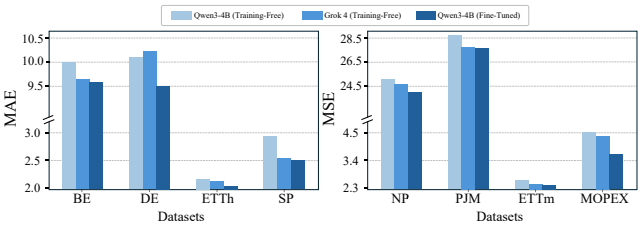


Fig. 9 Forecasting performance of training-free Grok 4, training-free Qwen3-4B, and trained Qwen3-4B in the evaluated configurations. Training adapts the forecasting model; Grok 4 remains the frozen general-purpose model in CastFlow

specialized forecasting response. The displayed narrative describes peak and timing adjustments, illustrating how a numerical prior and temporal diagnostics can be combined in a forecasting explanation. This example concerns the intended information flow and a selected forecast; it does not independently establish the faithfulness of the generated rationale.

5.5 Outcome-Aligned Training

Training Stages We examine how the two training stages contribute to specialized forecast refinement. Figure 8 compares the full model with variants omitting SFT or RLVR. The full model achieves the lowest error for every plotted dataset–metric pair, covering MAE on BE, DE, ETTh, and SP, and MSE on NP, PJM, ETTm, and MOPEX. Removing either stage increases error, supporting the combined use of supervised initialization and forecasting-outcome optimization in the evaluated setting.

These stages act on the forecasting LLM. SFT provides examples of generating structured forecasting responses from the workflow context, while RLVR evaluates generated forecasts using numerical rewards. Their complementary effects are consistent with this division of training objectives: imitation supplies an initial response policy, and reward optimization further improves the resulting predictions. The planner and reflection model remain frozen, so these results concern adaptation of evidence-conditioned forecasting rather than learned tool selection.

Forecasting Model Adaptation Figure 9 compares the reported training-free Qwen3-4B and Grok 4 configurations with Qwen3-4B after SFT and RLVR. Grok 4 achieves lower error than training-free Qwen3-4B on every plotted comparison except DE, indicating stronger training-free forecasting performance in most of the evaluated settings. After specialized training, Qwen3-4B achieves lower error than both training-free configurations for every plotted dataset–metric pair. The pattern appears across electricity-price, renewable-generation, and streamflow tasks, rather than being confined to one application.

Table 5 Reward-design comparison for the specialized forecasting model. Hybrid MSE combines absolute forecasting accuracy with improvement over the ensemble prior. Abs. and Rel. denote absolute and relative rewards. Best reported values are bolded.

Data	Metric	Abs. MSE	Rel. MSE	Hybrid MSE	Abs. MAE
BE	MSE	549.38	551.78	**546.87**	552.55
	MAE	9.68	9.61	**9.60**	9.62
DE	MSE	201.54	202.85	**200.47**	205.13
	MAE	**9.49**	9.54	9.50	9.80
SP	MSE	17.26	17.07	**16.90**	17.19
	MAE	2.52	2.52	**2.50**	2.52
ETTh	MSE	8.09	8.10	**8.00**	8.02
	MAE	2.08	2.08	**2.03**	2.08

The results support specialized adaptation as a practical route to improving the evaluated forecasting configurations. They also distinguish the numerical forecasting role from the general-purpose reasoning role: a compact model can produce more accurate forecasts after task-specific training, while the frozen Grok 4 backbone continues to support planning and reflection in CastFlow. We interpret this figure as a configuration-level comparison; the training-stage ablation provides the more direct comparison of retaining or removing SFT and RLVR.

Reward Formulation Table 5 compares absolute MSE, relative MSE, hybrid MSE, and absolute MAE rewards. The hybrid MSE formulation achieves the lowest MSE on all four reported datasets and the lowest MAE on BE, SP, and ETTh. On DE, absolute MSE yields a slightly lower MAE, 9.49 versus 9.50, while hybrid MSE gives the lowest MSE, 200.47. The benefit is therefore broad across the reported comparisons, with a small metric-specific exception.

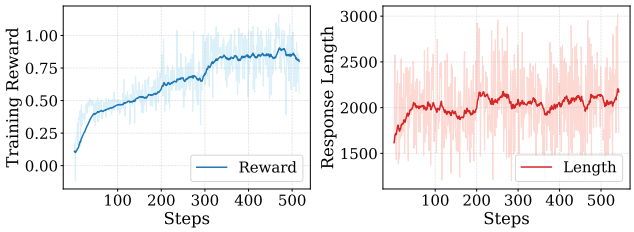


Fig. 10 GRPO training dynamics of the specialized forecaster. The reward reflects numerical accuracy and output validity; average response length records generated text length

The two terms evaluate complementary aspects of the scored forecasting output. The absolute term favors accurate predictions over the full horizon; the relative term evaluates improvement against the ensemble prior for the same instance. Their combination aligns the specialized forecaster with both final prediction quality and the utility of its refinement. Neither term alone matches the hybrid formulation across the reported results. Absolute MAE also produces higher MSE on each of the four datasets. These observations support the selected reward for forecasting refinement under the evaluated protocol, where offline targets score the numerical output and only the specialized model receives policy updates.

Optimization Dynamics Figure 10 shows the local model's GRPO training reward and response length. Reward rises rapidly early in training and subsequently increases more gradually, with fluctuations around the later, higher level. Average response length initially grows and then fluctuates within a comparatively stable range. Thus, the observed reward improvement does not require a continuing increase in generated text length.

Because the reward incorporates structural validity as well as numerical forecasting performance, the curve describes progress under the composite training objective. The stage ablation and forecasting comparisons provide complementary evidence about prediction accuracy. Response length characterizes the generated output.

5.6 Forecasting Backbone Analysis

We compare specialized forecasting backbones while retaining Grok 4 as the frozen reasoning model. Figure 11 shows per-dataset MSE ratios for Qwen3-1.7B, Qwen3-4B, Qwen3-8B, Llama-3.2-3B [77], and Mistral-7B [78]. For backbone m and dataset d, the normalization is $r_{m,d} = \mathrm{MSE}_{m,d}/\mathrm{MSE}_{\text{Qwen3-4B},d}$ within this comparison. The reference row is therefore one, and lower ratios indicate better forecasting performance on that dataset. We interpret these ratios separately for each dataset.

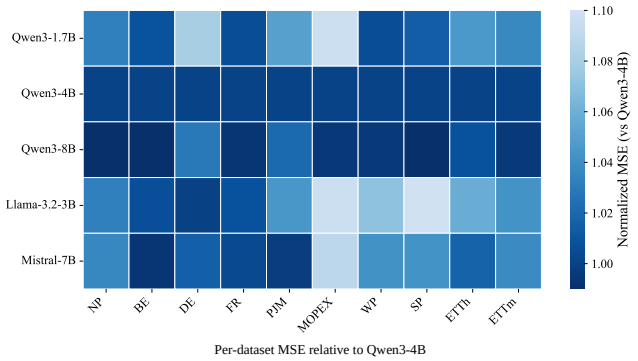


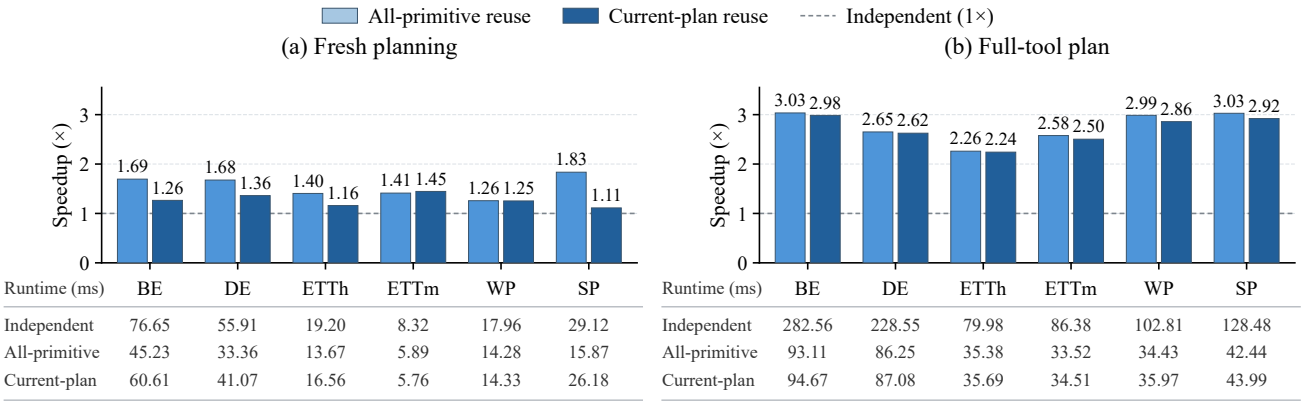
Fig. 11 Per-dataset forecasting backbone comparison with the frozen Grok 4 planner retained. Each cell reports MSE relative to Qwen3-4B on the same dataset; a ratio below one indicates lower error. The source per-dataset heatmap is retained without cross-dataset aggregation

The Qwen3-1.7B variant generally has higher error than the 4B reference, while increasing capacity to 8B does not improve every dataset: the 8B variant has higher relative MSE on DE and ETTh. Llama-3.2-3B and Mistral-7B are competitive on individual datasets, with relative errors varying across domains. Qwen3-4B remains a practical specialized forecaster because it is competitive across the displayed tasks while using fewer parameters than the 8B variant. This comparison supports the selected backbone under the evaluated configurations without requiring an aggregate error across datasets with different target scales.

5.7 System Efficiency and Execution Recovery

We evaluate the computational efficiency of diagnostic feature reuse and persistent stage recovery. The two experiments examine repeated feature computation within an action plan and forecast regeneration after interruption, respectively. They measure execution optimizations separately from the forecasting comparisons above.

Diagnostic Feature Reuse To evaluate the computational benefits of diagnostic feature reuse, we compare independent computation, all-primitive reuse, and current-plan reuse under fresh planning and a fixed full-tool plan. The evaluation uses ten fixed windows per dataset, separate from those used to develop the caching policy, with identical inputs and tool sequences across implementations to isolate the computational effects of feature reuse. Runtime is averaged over three repetitions after warm-up, with rotated execution order on a single CPU thread. Measurements cover diagnostic computation only and include all recorded plans, even those that invoke none of the shared tools.



Runtime (ms)	BE	DE	ETTh	ETTm	WP	SP
Independent	76.65	55.91	19.20	8.32	17.96	29.12
All-primitive	45.23	33.36	13.67	5.89	14.28	15.87
Current-plan	60.61	41.07	16.56	5.76	14.33	26.18

Runtime (ms)	BE	DE	ETTh	ETTm	WP	SP
Independent	282.56	228.55	79.98	86.38	102.81	128.48
All-primitive	93.11	86.25	35.38	33.52	34.43	42.44
Current-plan	94.67	87.08	35.69	34.51	35.97	43.99

Fig. 12 Diagnostic feature reuse under (a) fresh planning and (b) a full-tool plan. Bars show speedup relative to independent computation (dashed line). Aligned rows report runtime in milliseconds, averaged over ten windows and three repetitions. All implementations produce identical diagnostic outputs

As illustrated in Fig. 12, both reuse policies reduce mean diagnostic runtime across the evaluated datasets while preserving identical outputs. Specifically, current-plan reuse achieves speedups of 2.24–2.98 under full-tool plans and 1.11–1.45 under fresh planning. The larger gains under full-tool plans reflect greater overlap among diagnostic computations that would otherwise be repeated across tools. All-primitive reuse is generally faster, whereas current-plan reuse reduces the maximum serialized cache payload from approximately 14.5 kB to 8.9 kB. These findings demonstrate that current-plan reuse retains substantial computational savings while limiting the storage of intermediate features.

Recovery after a Controlled Interruption To evaluate the effectiveness of persistent stage recovery, we compare restarting a request with resuming its committed stages on six fixed windows from the training split of each of six datasets (BE, DE, ETTh, ETTm, WP, and SP). Each pair uses identical inputs, a fixed full-tool plan, deterministic validation, and the same historical ensemble and dataset-specific Qwen3-4B checkpoint trained with SFT and RLVR. These checkpoints differ from the jointly trained model used in the main forecasting experiment. We consider normal execution and an exception injected after forecast commitment but before validation, followed by journal reopening. This interruption point isolates whether completed forecasting work can be retained when the workflow resumes. Inference uses greedy decoding with a 5,000-token limit, and executor order alternates across pairs to reduce potential order effects. Timing includes journal I/O and excludes model-service initialization; all attempts, including four ETTm format-validation failures, remain in the timing and call statistics.

As shown in Table 6, the two executors exhibit similar mean runtimes under normal execution. Under interruption, stage recovery reduces mean execution time across all six evaluated datasets. For the 35 pairs that reach the scheduled interruption point, recovery preserves identical final predictions and halves forecast calls, achieving a median paired speedup of 1.98. These gains arise from resuming validation directly from the committed forecast, avoiding another invocation of the forecasting module. This benefit is especially relevant when forecast generation accounts for a substantial share of execution time, since retaining its output prevents that cost from being incurred again during request recovery. These results highlight how persistent execution state supports efficient workflow recovery.

6 Conclusion

We presented CastFlow, a role-specialized framework for forecast-relevant evidence acquisition and numerical refinement. Training-built Experience Memory guides a frozen LLM's analytical tool use, while a specialized forecaster refines an ensemble prior from the acquired diagnostics. SFT and verifiable forecasting rewards adapt only this forecaster. Across ten benchmarks, CastFlow achieves the lowest MSE and MAE among evaluated methods on nine datasets. Component and training analyses support the complete workflow while exposing dataset-dependent diagnostic utility and attribution limits. These findings motivate coordinating evidence acquisition with outcome-aligned numerical refinement in agentic time series forecasting.

Table 6 Mean execution time (seconds) for restart and stage recovery on six paired windows per dataset, under normal execution and a controlled post-forecast interruption. All attempts, including failed forecasts, remain in the averages. Timing includes journal I/O and excludes model-service initialization. Bold indicates the lower mean within each condition.

Condition	Executor	BE	DE	ETTh	ETTm	WP	SP
Normal	Restart	15.02	17.20	**40.48**	**41.89**	39.75	41.54
	Resume	**14.77**	**16.72**	40.74	42.67	**38.90**	**40.91**
Interrupted	Restart	29.01	34.30	80.41	77.30	76.22	82.56
	Resume	**14.80**	**16.82**	**40.44**	**46.28**	**38.35**	**46.67**

A Diagnostic Tool Definitions

This appendix collects the diagnostic computations used by the toolkit in Section 4.2. Their outputs characterize the observed lookback window and available covariates; they do not require the current request's future forecasting targets.

A.1 Statistical and Spectral Profiler

The statistical analysis tool returns the mean μ, standard deviation σ, and boundary extrema. The basic statistics tool includes robust dispersion measures such as

$$\mathrm{MAD}(\mathbf{x}) = \mathrm{median}(|x_t - \tilde{x}|), \tag{16}$$

where $\tilde{x}$ is the sequence median. Spectral entropy characterizes the dispersion of spectral power:

$$S_{\mathrm{spec}}(\mathbf{x}) = -\sum_{k=1}^{\lfloor L/2 \rfloor} P_k \log P_k, \tag{17}$$

where

$$P_k = \frac{|\mathcal{F}(\mathbf{x})_k|^2}{\sum_{q=1}^{\lfloor L/2 \rfloor} |\mathcal{F}(\mathbf{x})_q|^2}. \tag{18}$$

Here, $\mathcal{F}$ denotes the discrete Fourier transform. Greater entropy corresponds to a more dispersed spectrum. The data quality tool measures dropout ratios and supplies the range diagnostic $\mathcal{B} = [\mu - \kappa\sigma, \mu + \kappa\sigma]$. The comprehensive feature tool aggregates these outputs as $\mathcal{S}_{\mathrm{stat}} = \langle \mu, \sigma, \mathrm{MAD}, S_{\mathrm{spec}}, \mathcal{B} \rangle$. These quantities provide descriptive evidence for the forecaster and verifier.

A.2 Dynamics Monitor

The trend analysis tool computes the linear slope

$$m = \frac{\sum_t (t - \bar{t})(x_t - \bar{x})}{\sum_t (t - \bar{t})^2}. \tag{19}$$

The changepoint trend tool examines the first and second differences, $\Delta x_t = x_t - x_{t-1}$ and $\Delta^2 x_t = \Delta x_t - \Delta x_{t-1}$, as diagnostics of changing local dynamics. The cross-channel and exogenous analysis tools examine lagged associations. For an observed target sequence x and auxiliary sequence u, the normalized lagged correlation is

$$\rho_{x,u}(d) = \frac{\sum_t (x_t - \bar{x})(u_{t+d} - \bar{u})}{\sqrt{\sum_t (x_t - \bar{x})^2 \sum_t (u_{t+d} - \bar{u})^2}}, \tag{20}$$

where d denotes the lag and the sums use the available paired observations. The event summary tool complements these numerical features with a dominant qualitative label in $\{\mathrm{rise}, \mathrm{fall}, \mathrm{flat}, \mathrm{oscillation}\}$.

A.3 Residual Diagnoser

The autoregressive residual tool fits a proxy process on the observed lookback window and computes

$$\epsilon_t = x_t - \left(c + \sum_{i=1}^{p} \phi_i x_{t-i} \right), \tag{21}$$

where p is selected by information criteria. It summarizes residual bias and first-order dependence as

$$\mu_\epsilon = \frac{1}{L-p} \sum_{t=p+1}^{L} \epsilon_t, \qquad r_1 = \frac{\sum_{t=p+2}^{L} \epsilon_t \epsilon_{t-1}}{\sum_{t=p+1}^{L} \epsilon_t^2}. \tag{22}$$

This tool is available only during training and is bypassed at inference. Its lookback-fitted residuals describe an autoregressive proxy, not the unknown future errors of the ensemble forecasting prior.

B Workflow Execution Support

Experience Memory provides historical procedural guidance. The following mechanisms reuse computations or committed stages within the execution of a forecasting request. They support the efficiency and recovery experiments in Section 5.7.

B.1 Diagnostic Feature Reuse

Several diagnostic tools share computations such as statistics, temporal differences, and cross-channel summaries. A request-local feature cache avoids recomputing identical operations. Each key identifies the primitive operation, its input values and array structure, and its extraction parameters, including the seasonal period. A missing result is computed once and a copy is returned on subsequent matching calls. Each request starts with an empty cache; computations involving mutable fitted reference profiles are excluded.

The all-primitive policy caches every eligible primitive. The current-plan policy instead caches primitives referenced more than once by the current tool plan, using dependency counts from tool definitions and updating eligibility when the plan changes. It uses the current plan without inspecting later retry plans or forecasting targets. Both policies preserve tool interfaces and diagnostic values. Their benefit depends on actual overlap: cache lookup and copying can offset savings when a plan contains few repeated computations.

Table 7 Tool inventory, inputs, outputs, and execution scope.

Tool	Input	Main output	Invocation / phase
Foundational anchorer (1 tool)			
Model auxiliary	Lookback; historical expert library	Ensemble forecast prior $\hat{\mathbf{y}}_{\text{base}}$	Mandatory; training and inference
Statistical and spectral profiler (4 tools)			
Statistical analysis	Target lookback	Mean, standard deviation, extrema, and range	Optional; training and inference
Basic statistics	Target lookback	Median, MAD, and spectral entropy	Optional; training and inference
Data quality	Lookback values and missingness	Dropout ratio; bounds $[\mu - \kappa\sigma, \mu + \kappa\sigma]$	Optional; training and inference
Comprehensive feature	Target lookback	Aggregated statistical, spectral, and quality diagnostics	Optional; training and inference
Dynamics monitor (5 tools)			
Trend analysis	Target lookback	Linear slope and trend direction	Optional; training and inference
Changepoint trend	Target lookback	First- and second-difference summaries	Optional; training and inference
Cross-channel	Target and auxiliary lookback	Historical correlations and lead–lag associations	Optional; training and inference
Exogenous analysis	Target and auxiliary lookback	Historical target–auxiliary correlations	Mandatory; training and inference
Event summary	Target lookback	Rise, fall, flat, or oscillation label	Optional; training and inference
Residual diagnoser (1 tool)			
Autoregressive residual	Target lookback	Lookback-fitted residual mean and lag-one autocorrelation	Optional; training only

Inputs must be available at the forecast origin. Horizon covariate forecasts $\mathbf{u}^{[t]}$ enter the forecasting context through the supported interface; historical associations use lookback observations. Optional tools are selected by the frozen planner. The residual tool is skipped at inference; archived residuals describe training windows. Diagnostic features may overlap across categories. MAD denotes median absolute deviation.

B.2 Persistent Stage Recovery

Forecast revision changes evidence or prompt context to generate a new response. Execution recovery resumes completed work after an interruption with the same forecasting inputs, including any supplied horizon covariate forecasts. A persistent execution journal records completed planning, tool, forecasting, and validation stages. A request identity binds the forecasting inputs, model configuration, generation settings, and implementation version. Stage records contain an input fingerprint, the output, and an output hash. A completed stage is reused only when the request identity and stage inputs match and integrity checks pass. Thus, a request interrupted after forecast commitment can resume validation without regenerating that forecast.

The journal uses a request lock and atomic file replacement with explicit synchronization. Semantic revisions receive new plan and forecast stage identifiers, while previous responses and rejected outputs remain recorded. Only tools declared reusable retain stage identifiers across revisions with unchanged inputs. New requests never inherit another request's numerical predictions. An external call interrupted before durable commitment may need to be repeated; recovery therefore does not provide exactly-once execution of external services. The reported evaluation isolates diagnostic computation and a controlled post-forecast interruption, rather than arbitrary service failures or end-to-end scalability.

C Workflow Decision-Process Interpretation

The collaborative workflow can be described by a decision process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$. Its state contains the observed lookback, origin-issued horizon covariate forecasts, current ensemble prior, retrieved experience, and workflow history:

$$s_j = (\mathbf{x}, \mathbf{u}^{[t]}, \hat{\mathbf{y}}_{\text{base},j}, \mathcal{M}_{\text{ret}}, \mathcal{H}_j). \tag{23}$$

The prior is empty until tool execution produces it. Planning actions select tools, and forecasting actions produce numerical refinements through token generation. Tool execution and the addition of observations and feedback induce transitions. A full trajectory $\tau = (s_1, a_1, \ldots, s_M, a_M)$ is generated by the combined workflow policy $\Pi_{\Theta,\theta}$, comprising frozen planning and reflection, deterministic execution, and the trainable forecasting response policy π_θ. The training objective in Eq. 15 updates only θ.

An idealized risk argument motivates retaining multiple diagnostic observations. Let $\mathcal{O}_m = \{o_1, \ldots, o_m\}$ be the observations collected after m interactions and define

$$R^\star_m = \inf_{f \in \mathcal{F}} \mathbb{E}[\ell(\mathbf{y}, f(\mathbf{x}, \mathbf{u}^{[t]}, \mathcal{O}_m))]. \tag{24}$$

If the predictor family can ignore additional observations, then $R^\star_{m+1} \leq R^\star_m$ for leakage-free tool execution. This is a property of the idealized predictor family, not a guarantee that a finite learned model improves after every interaction. Deterministic diagnostics also need not add external information beyond the origin-available inputs $(\mathbf{x}, \mathbf{u}^{[t]})$; they provide alternative representations that can support the forecasting model.

D Reproducibility and Information Availability

D.1 Forecast-Origin Information Boundary

Equation 2 defines admissible source information $\mathcal{I}_t$ at forecast origin t. The inference protocol supplies the observed lookback and horizon-aligned covariate forecasts $\mathbf{u}^{[t]}$ issued by t; diagnostics use admissible inputs and fixed offline resources. An issued forecast can describe a future period, whereas its realized observation is unavailable at t. Supplied forecasts remain fixed through reflection retries, excluding later revisions. Admissibility therefore depends on provenance and release time, not merely the time described by a value.

Test Experience Memory contains training experiences and remains read-only; training targets support offline trajectory selection. The current test target is used only after forecasting for evaluation, never for retrieval, planning, reflection, or memory maintenance. Request history $\mathcal{H}_j$ may change through retries. Archived diagnostics remain examples from their training windows, not observations of the test window.

D.2 Reported Experimental Configuration

Table 8 summarizes the reported settings. The anchorer expert pool, comparison baselines, and forecasting LLM are separate configurations; historical expert-weighting losses and response-scoring MSE also serve different roles.

D.3 Configuration Scope

The GRPO sampling settings describe training, whereas greedy decoding and dataset-specific checkpoints apply to the recovery experiment in Section 5.7.

D.4 Prompt and Response Interfaces

Planning receives the observed window, tool descriptions, mandatory-tool requirements, and retrieved guidance, and returns a tool schedule. Forecasting receives observations, $\mathbf{u}^{[t]}$, the ensemble prior, diagnostics, procedural guidance, and revision feedback, and returns reasoning plus horizon-wide predictions. Memory therefore guides both planning and forecasting; archived observations and answers remain training examples. Relevant tools consume covariate forecasts through their supported interfaces. Reflection first checks structural validity, then evidence consistency without test targets; semantic acceptance does not guarantee numerical accuracy.

Public prompt examples are in `forecasting/modules.py` and `forecasting/prompts.py` at revision `c6a993d681f9`.

Table 8 Reported settings for the main forecasting evaluation. Memory is fixed throughout test inference.

Component	Setting
General-purpose model	Grok 4; planning and reflection frozen
Specialized model	Qwen3-4B; forecasting parameters updated
SFT	Learning rate 5×10^{-5}; batch size 8; 1 epoch
RLVR	GRPO; learning rate 2×10^{-6}; 3 epochs
GRPO sampling	Group size 8; temperature 1.0; KL coefficient 0
GRPO scoring	Direct response-level reward; no candidate-wise reflection loop
Completion limit	5,000 tokens for the trainable forecasting model
Short-term tasks	Lookback 168; forecast horizon 24
Long-term tasks	Lookback 96; forecast horizon 96
Data partition	Chronological training/validation/test ratio 7:1:2
Horizon covariates	Auxiliary forecasts issued no later than the forecast origin
Test Experience Memory	Training-built library; retrieval only
Evaluation scale	MSE and MAE on the original target scale

D.5 Training Samples and Inference Outputs

SFT imitates teacher responses in selected pairs $(\mathcal{C}, r^*)$; memory selection instead retains historical procedures. RLVR samples responses sharing a context, diagnostics, covariate forecasts, and prior, and scores each numerical answer directly. Groupwise advantages train only specialized forecasting-response tokens, without candidate-wise reflection or retries or gradients to planning and reflection tokens.

If the retry budget is exhausted without acceptance, fallback uses a structurally valid request-history response, then the ensemble prior; an output is missing or abnormal only if neither is usable. Fallback never selects by test error.

D.6 Reward Configuration

Equations 13 and 12 define the absolute and relative rewards, with $\lambda = 5$, $\delta = 0.5$, and Table 9's fixed MSE scales. Here γ_d controls the absolute mapping and ν_d scales prior-minus-candidate MSE. Without a usable matched prior, only the absolute term is returned.

Table 9 Dataset-specific reward scales (main configuration).

Dataset	Absolute scale γ_d	Relative scale ν_d
BE	800	10
DE	350	10
FR	1000	10
NP	40	3
PJM	45	4
ETTh	12	2
ETTm	3.5	1
WP	3000	30
SP	50	3
MOPEX	7	1.5

Before numerical scoring, an unparseable nonempty answer receives -1; an empty or shorter-than-H answer receives -0.8. For $n > H$ values, the reward is

$$\max\left(-0.5 - 0.6\frac{n-H}{H}, -0.8\right). \tag{25}$$

A missing or unparseable target or NaN candidate MSE receives -1; NaN baseline MSE disables only the relative term. Numerical training reward does not require semantic reflection acceptance. The reference parser extracts prediction and target sequences from supported keys, and the horizon-wide losses in Section 4.4 assume target-aligned predictions.

D.7 Dataset Provenance and Access

The five EPF markets originate from the benchmark of Lago et al. [66]; this study uses the TimeXer EPF collection [67]. ETT data are provided through the authors' ETDataset repository. The renewable-power datasets originate from the iFLYTEK challenge cited in the main text; access remains subject to the competition platform. MOPEX provenance is described by Schaake et al. [2]; the original data description and its institutional report archive document the source and collection procedure.

The experimental settings specify chronological splits, targets, horizons, and metric scale. Study-specific processed windows and split manifests are not publicly archived, as stated in Data availability.

Statements and Declarations

Funding This research was supported by grants from the National Natural Science Foundation of China (No. 62502486, 62337001), the grants of Provincial Natural Science Foundation of Anhui Province (No. 2408085QF193), the Fundamental Research Funds for the Central Universities of China (No. WK2150110032).

Competing interests The authors have no relevant financial or non-financial interests to disclose.

Data availability The sources and access conditions for the benchmark datasets used in this study are documented in Appendix D.7. CastFlow-specific processed windows and exact split manifests are not publicly archived. Requests for study-specific prepared data should be directed to the corresponding author and will be considered subject to the original providers' access and redistribution conditions.

Code availability A reference implementation of CastFlow is publicly available at https://github.com/Forever-Pan/CastFlow. The archived public revision and its reproducibility scope are documented in Appendix D.

References

1. iFLYTEK AI Challenge: 2025 iflytek renewable power forecasting challenge (wind and solar). https://challenge.xfyun.cn/topic/info?type=renewable-power-forecast\&option=ssgy\&ch=dwsf259 (2025). Accessed: Apr. 30, 2026
2. Schaake, J., Cong, S., Duan, Q.: The US MOPEX data set. In: Large Sample Basin Experiments for Hydrological Model Parameterization: Results of the Model Parameter Experiment (MOPEX), *IAHS Publication*, vol. 307, pp. 9–28. IAHS Press (2006). URL https://iahs.info/uploads/dms/13600.04-9-28-SCHAAKE.pdf
3. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: International Conference on Learning Representations (2018). URL https://openreview.net/forum?id=SJiHXGWAZ
4. Zhao, L., Shen, Y.: Rethinking channel dependence for multivariate time series forecasting: Learning from leading indicators. In: International Conference on Learning Representations (2024). URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/b52b07a239a7afa155ca25cf17a55074-Abstract-Conference.html
5. Benidis, K., Rangapuram, S.S., Flunkert, V., Wang, Y., Maddix, D.C., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., et al.: Deep learning for time series forecasting: Tutorial and literature survey. ACM Computing Surveys **55**(6), 1–36 (2022). https://doi.org/10.1145/3533382
6. Cheng, M., Liu, Z., Tao, X., Liu, Q., Zhang, J., Pan, T., Zhang, S., He, P., Zhang, X., Wang, D., Wang, J., Chen, E.: A comprehensive survey of time series forecasting: Concepts, challenges, and future directions. IEEE Transactions on Knowledge and Data Engineering pp. 1–20 (2026). https://doi.org/10.1109/TKDE.2026.3736707. Early access
7. Qiu, X., Hu, J., Zhou, L., Wu, X., Du, J., Zhang, B., Guo, C., Zhou, A., Jensen, C.S., Sheng, Z., Yang, B.: TFB: Towards comprehensive and fair benchmarking of time series forecasting methods. Proceedings of the VLDB Endowment **17**(9), 2363–2377 (2024). https://doi.org/10.14778/3665844.3665863
8. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K.R., Cao, Y.: ReAct: Synergizing reasoning and acting in language models. In: International Conference on Learning Representations (2023). URL https://openreview.net/forum?id=WE_vluYUL-X
9. Cheng, M., Tao, X., Liu, Q., Guo, Z., Chen, E.: Position: Beyond model-centric prediction – agentic time series forecasting. arXiv preprint arXiv:2602.01776 (2026). URL https://arxiv.org/abs/2602.01776

10. Fischer, U., Dannecker, L., Siksnys, L., Rosenthal, F., Boehm, M., Lehner, W.: Towards integrated data analytics: Time series forecasting in DBMS. Datenbank-Spektrum **13**(1), 45–53 (2013). https://doi.org/10.1007/s13222-012-0108-4
11. Januschowski, T., Wang, Y., Gasthaus, J., Rangapuram, S., Türkmen, C., Zschiegner, J., Stella, L., Bohlke-Schneider, M., Maddix, D., Benidis, K., Alexandrov, A., Faloutsos, C., Schelter, S.: A flexible forecasting stack. Proceedings of the VLDB Endowment **17**(12), 3883–3892 (2024). https://doi.org/10.14778/3685800.3685813
12. Cheng, M., Wang, J., Wang, D., Tao, X., Liu, Q., Chen, E.: Can slow-thinking LLMs reason over time? empirical studies in time series forecasting. In: Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining, pp. 99–110 (2026). https://doi.org/10.1145/3773966.3777931
13. Zhou, Y., Luo, Y., Cheng, M., Liu, Q., Wang, J., Wang, D., Chen, E.: Time series forecasting via reasoning: A slow-thinking approach with reinforcement fine-tuned LLMs. In: ACM International Conference on Information and Knowledge Management (CIKM) (2026). URL https://arxiv.org/abs/2506.10630. Accepted for publication
14. Tao, X., Cheng, M., Guo, Z., Yu, S., Liu, Y., Liu, Q., Wang, S.: MemCast: Memory-driven time series forecasting with experience-conditioned reasoning. In: Proceedings of the 43rd International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 306, pp. 119,553–119,569. PMLR (2026). URL https://proceedings.mlr.press/v306/tao26a.html
15. Zhao, H., Zhang, X., Wei, J., Xu, Y., He, Y., Sun, S., You, C.: TimeSeriesScientist: A general-purpose AI agent for time series analysis. arXiv preprint arXiv:2510.01538 (2025). URL https://arxiv.org/abs/2510.01538
16. Zhang, X., Gao, T., Cheng, M., Pan, B., Guo, Z., Liu, Y., Tao, X., Liu, Q.: AlphaCast: A human wisdom-LLM intelligence co-reasoning framework for time series forecasting. In: ACM International Conference on Information and Knowledge Management (CIKM) (2026). URL https://arxiv.org/abs/2511.08947. Accepted for publication
17. Tao, X., Cheng, M., Jiang, C., Gao, T., Zhang, H., Liu, Y.: Cast-R1: Learning tool-augmented sequential decision policies for time series forecasting. arXiv preprint arXiv:2602.13802 (2026). URL https://arxiv.org/abs/2602.13802
18. Fang, Y., Miao, H., Liang, Y., Deng, L., Cui, Y., Zeng, X., Xia, Y., Zhao, Y., Pedersen, T.B., Jensen, C.S., Zhou, X., Zheng, K.: Unraveling spatio-temporal foundation models via the pipeline lens: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering **38**(3), 2040–2063 (2026). https://doi.org/10.1109/TKDE.2026.3651536
19. Hyndman, R.J., Khandakar, Y.: Automatic time series forecasting: The forecast package for R. Journal of Statistical Software **27**(3), 1–22 (2008). https://doi.org/10.18637/jss.v027.i03
20. Gardner Jr., E.S.: Exponential smoothing: The state of the art. Journal of Forecasting **4**(1), 1–28 (1985). https://doi.org/10.1002/for.3980040103
21. Wang, Y., Wu, H., Dong, J., Liu, Y., Wang, C., Long, M., Wang, J.: Deep time series models: A comprehensive survey and benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence **48**(10), 11,642–11,662 (2026). https://doi.org/10.1109/TPAMI.2026.3690845
22. Sapankevych, N.I., Sankar, R.: Time series prediction using support vector machines: A survey. IEEE Computational Intelligence Magazine **4**(2), 24–38 (2009). https://doi.org/10.1109/MCI.2009.932254
23. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016). https://doi.org/10.1145/2939672.2939785
24. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: LightGBM: A highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, vol. 30, pp. 3146–3154 (2017). URL https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html
25. Bontempi, G., Ben Taieb, S., Le Borgne, Y.A.: Machine learning strategies for time series forecasting. In: Business Intelligence: Second European Summer School, eBISS 2012, Brussels, Belgium, July 15–21, 2012, Tutorial Lectures, *Lecture Notes in Business Information Processing*, vol. 138, pp. 62–77. Springer (2013). https://doi.org/10.1007/978-3-642-36318-4_3
26. Li, B., Luo, Y., Liu, Z., Zheng, J., Lv, J., Ma, Q.: HyperIMTS: Hypergraph neural network for irregular multivariate time series forecasting. In: Proceedings of the 42nd International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 267, pp. 35,502–35,518. PMLR (2025). URL https://proceedings.mlr.press/v267/li25bl.html
27. Huang, Q., Zhou, Z., Yang, K., Yi, Z., Wang, X., Wang, Y.: TimeBase: The power of minimalism in efficient long-term time series forecasting. In: Proceedings of the 42nd International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 267, pp. 26,227–26,246. PMLR (2025). URL https://proceedings.mlr.press/v267/huang25az.html
28. Eldele, E., Ragab, M., Chen, Z., Wu, M., Li, X.: TSLANet: Rethinking transformers for time series representation learning. In: Proceedings of the 41st International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 235, pp. 12,409–12,428. PMLR (2024). URL https://proceedings.mlr.press/v235/eldele24a.html
29. Chen, Y., Céspedes, N., Barnaghi, P.: A closer look at transformers for time series forecasting: Understanding why they work and where they struggle. In: Proceedings of the 42nd International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 267, pp. 7763–7780. PMLR (2025). URL https://proceedings.mlr.press/v267/chen25f.html
30. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: Proceedings of the 39th International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 162, pp. 27,268–27,286. PMLR (2022). URL https://proceedings.mlr.press/v162/zhou22g.html
31. Ahamed, M.A., Cheng, Q.: TimeMachine: A time series is worth 4 mambas for long-term forecasting. In: ECAI 2024: 27th European Conference on Artificial Intelligence, Including 13th Conference on Prestigious Applications of Intelligent Systems, PAIS 2024, Proceedings, *Frontiers in Artificial Intelligence and Applications*, vol. 392, pp. 1688–1695. IOS Press (2024). https://doi.org/10.3233/FAIA240677
32. Challu, C., Olivares, K.G., Oreshkin, B.N., Garza Ramirez, F., Mergenthaler Canseco, M., Dubrawski, A.: NHITS: Neural hierarchical interpolation for time series forecasting. Proceedings of the AAAI Conference on Artificial

Intelligence **37**(6), 6989–6997 (2023). https://doi.org/10.1609/aaai.v37i6.25854
33. Woo, G., Liu, C., Sahoo, D., Kumar, A., Hoi, S.: ETSformer: Exponential smoothing transformers for time-series forecasting. In: International Conference on Learning Representations (2023). URL https://openreview.net/forum?id=5m_3whfo483
34. Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., Long, M.: iTransformer: Inverted transformers are effective for time series forecasting. In: International Conference on Learning Representations (2024). URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/2ea18fdc667e0ef2ad82b2b4d65147ad-Abstract-Conference.html
35. Cheng, M., Yang, J., Pan, T., Liu, Q., Li, Z., Wang, S.: ConvTimeNet: A deep hierarchical fully convolutional model for multivariate time series analysis. In: Companion Proceedings of the ACM on Web Conference 2025, pp. 171–180 (2025). https://doi.org/10.1145/3701716.3715214
36. Ansari, A.F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S.S., Arango, S.P., Kapoor, S., Zschiegner, J., Maddix, D.C., Wang, H., Mahoney, M.W., Torkkola, K., Wilson, A.G., Bohlke-Schneider, M., Wang, Y.: Chronos: Learning the language of time series. Transactions on Machine Learning Research (2024). URL https://openreview.net/forum?id=gerNCVqqtR
37. Das, A., Kong, W., Sen, R., Zhou, Y.: A decoder-only foundation model for time-series forecasting. In: Proceedings of the 41st International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 235, pp. 10,148–10,167. PMLR (2024). URL https://proceedings.mlr.press/v235/das24c.html
38. Liu, Y., Qin, G., Shi, Z., Chen, Z., Yang, C., Huang, X., Wang, J., Long, M.: Sundial: A family of highly capable time series foundation models. In: Proceedings of the 42nd International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 267, pp. 39,295–39,317. PMLR (2025). URL https://proceedings.mlr.press/v267/liu25be.html
39. Shi, H.N., Huang, T.J., Han, L., Zhan, D.C., Ye, H.J.: One-embedding-fits-all: Efficient zero-shot time series forecasting by a model zoo. arXiv preprint arXiv:2509.04208 (2025). URL https://arxiv.org/abs/2509.04208
40. Zhang, X., Chowdhury, R.R., Gupta, R.K., Shang, J.: Large language models for time series: A survey. In: K. Larson (ed.) Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24, pp. 8335–8343. International Joint Conferences on Artificial Intelligence Organization (2024). https://doi.org/10.24963/ijcai.2024/921. Survey Track
41. Jiang, Y., Pan, Z., Zhang, X., Garg, S., Schneider, A., Nevmyvaka, Y., Song, D.: Empowering time series analysis with large language models: A survey. In: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24, pp. 8095–8103. International Joint Conferences on Artificial Intelligence Organization (2024). https://doi.org/10.24963/ijcai.2024/895. Survey Track
42. Liu, C., Xu, Q., Miao, H., Yang, S., Zhang, L., Long, C., Li, Z., Zhao, R.: TimeCMA: Towards LLM-empowered multivariate time series forecasting via cross-modality alignment. Proceedings of the AAAI Conference on Artificial Intelligence **39**(18), 18,780–18,788 (2025). https://doi.org/10.1609/aaai.v39i18.34067
43. Qiu, S., Du, L., Zeng, C., Deng, L., Luo, Q., Tang, J.: EAST: Embedding-aligned modality adaptation for large language model-based traffic forecasting. In: Web and Big Data, *Lecture Notes in Computer Science*, vol. 16995, pp. 67–82. Springer (2027). https://doi.org/10.1007/978-981-92-5699-0_5
44. Xue, H., Salim, F.D.: PromptCast: A new prompt-based learning paradigm for time series forecasting. IEEE Transactions on Knowledge and Data Engineering **36**(11), 6851–6864 (2024). https://doi.org/10.1109/TKDE.2023.3342137
45. Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J.Y., Shi, X., Chen, P.Y., Liang, Y., Li, Y.F., Pan, S., Wen, Q.: Time-LLM: Time series forecasting by reprogramming large language models. In: International Conference on Learning Representations (2024). URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/680b2a8135b9c71278a09cafb605869e-Abstract-Conference.html
46. Pan, Z., Jiang, Y., Garg, S., Schneider, A., Nevmyvaka, Y., Song, D.: S^2IP-LLM: Semantic space informed prompt learning with LLM for time series forecasting. In: Proceedings of the 41st International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 235, pp. 39,135–39,153. PMLR (2024). URL https://proceedings.mlr.press/v235/pan24c.html
47. Tao, X., Zhang, S., Cheng, M., Wang, D., Pan, T., Pan, B., Zhang, C., Wang, S.: From values to tokens: An LLM-driven framework for context-aware time series forecasting via symbolic discretization. In: Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, pp. 3043–3051 (2026). https://doi.org/10.24963/ijcai.2026/338
48. Gruver, N., Finzi, M., Qiu, S., Wilson, A.G.: Large language models are zero-shot time series forecasters. In: Advances in Neural Information Processing Systems, vol. 36, pp. 19,622–19,635 (2023). https://doi.org/10.52202/075280-0861
49. Jalori, G., Verma, P., Arik, S.O.: FLAIRR-TS: Forecasting LLM-agents with iterative refinement and retrieval for time series. In: Findings of the Association for Computational Linguistics: EMNLP 2025, pp. 15,427–15,437. Association for Computational Linguistics (2025). https://doi.org/10.18653/v1/2025.findings-emnlp.834
50. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., Scialom, T.: Toolformer: Language models can teach themselves to use tools. In: Advances in Neural Information Processing Systems, vol. 36, pp. 68,539–68,551 (2023). https://doi.org/10.52202/075280-2997
51. Wang, Z., Cai, S., Chen, G., Liu, A., Ma, X., Liang, Y.: Describe, explain, plan and select: Interactive planning with LLMs enables open-world multi-task agents. In: Advances in Neural Information Processing Systems, vol. 36, pp. 34,153–34,189 (2023). https://doi.org/10.52202/075280-1480
52. Shankar, S., Chambers, T., Shah, T., Parameswaran, A.G., Wu, E.: DocETL: Agentic query rewriting and evaluation for complex document processing. Proceedings of the VLDB Endowment **18**(9), 3035–3048 (2025). https://doi.org/10.14778/3746405.3746426
53. Li, Z., Zhao, W., Katsifodimos, A., Hai, R.: LLM-PQA: LLM-enhanced prediction query answering. In: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, pp. 5239–5243. ACM (2024). https://doi.org/10.1145/3627673.3679210
54. Tao, X., Wu, Y., Cheng, M., Guo, Z., Gao, T.: AnomaMind: Agentic time series anomaly detection with tool-augmented reasoning. arXiv preprint arXiv:2602.13807 (2026). URL https://arxiv.org/abs/2602.13807

55. Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., Gupta, S., Majumder, B.P., Hermann, K., Welleck, S., Yazdanbakhsh, A., Clark, P.: Self-Refine: Iterative refinement with self-feedback. In: Advances in Neural Information Processing Systems, vol. 36, pp. 46,534–46,594 (2023). https://doi.org/10.52202/075280-2019
56. Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: Language agents with verbal reinforcement learning. In: Advances in Neural Information Processing Systems, vol. 36, pp. 8634–8652 (2023). https://doi.org/10.52202/075280-0377
57. Tan, X., Wang, X., Liu, Q., Xu, X., Yuan, X., Zhu, L., Zhang, W.: MemoTime: Memory-augmented temporal knowledge graph enhanced large language model reasoning. In: Proceedings of the ACM Web Conference 2026, pp. 4220–4231. ACM (2026). https://doi.org/10.1145/3774904.3792581
58. Zelikman, E., Wu, Y., Mu, J., Goodman, N.: STaR: Bootstrapping reasoning with reasoning. In: Advances in Neural Information Processing Systems, vol. 35, pp. 15,476–15,488 (2022). https://doi.org/10.52202/068431-1126
59. Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z.F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., et al.: DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. Nature **645**(8081), 633–638 (2025). https://doi.org/10.1038/s41586-025-09422-z
60. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al.: Qwen3 technical report. arXiv preprint arXiv:2505.09388 (2025). URL https://arxiv.org/abs/2505.09388
61. Guan, T., Meng, Z., Li, D., Wang, S., Yang, C.H.H., Wen, Q., Liu, Z., Siniscalchi, S., Jin, M., Pan, S.: TimeOmni-1: Incentivizing complex reasoning with time series in large language models. In: International Conference on Learning Representations (2026). URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/f7227d037ee7202106f511f78e9ecb80-Abstract-Conference.html
62. Tang, Z., Yin, X., Chen, W., Chen, Z., Zheng, Y., Ye, W., Wang, K., Lin, L.: AlphaAgentEvo: Evolution-oriented alpha mining via self-evolving agentic reinforcement learning. In: International Conference on Learning Representations (2026). URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/741aab8b41a2987867acc9939ad50383-Abstract-Conference.html
63. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley (1990). https://doi.org/10.1002/9780470316801
64. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing **26**(1), 43–49 (1978). https://doi.org/10.1109/TASSP.1978.1163055
65. Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y.K., Wu, Y., Guo, D.: DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300 (2024). URL https://arxiv.org/abs/2402.03300
66. Lago, J., Marcjasz, G., De Schutter, B., Weron, R.: Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark. Applied Energy **293**, 116983 (2021). https://doi.org/10.1016/j.apenergy.2021.116983
67. Wang, Y., Wu, H., Dong, J., Qin, G., Zhang, H., Liu, Y., Qiu, Y., Wang, J., Long, M.: TimeXer: Empowering transformers for time series forecasting with exogenous variables. In: Advances in Neural Information Processing Systems, vol. 37, pp. 469–498 (2024). https://doi.org/10.52202/079017-0015
68. Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W.: Informer: Beyond efficient transformer for long sequence time-series forecasting. Proceedings of the AAAI Conference on Artificial Intelligence **35**(12), 11,106–11,115 (2021). https://doi.org/10.1609/aaai.v35i12.17325
69. Taylor, S.J., Letham, B.: Forecasting at scale. The American Statistician **72**(1), 37–45 (2018). https://doi.org/10.1080/00031305.2017.1380080
70. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In: Advances in Neural Information Processing Systems, vol. 34, pp. 22,419–22,430 (2021). URL https://proceedings.neurips.cc/paper/2021/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html
71. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? Proceedings of the AAAI Conference on Artificial Intelligence **37**(9), 11,121–11,128 (2023). https://doi.org/10.1609/aaai.v37i9.26317
72. Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: Long-term forecasting with transformers. In: International Conference on Learning Representations (2023). URL https://openreview.net/forum?id=Jbdc0vTOcol
73. xAI: Grok 4 Model Card. Model card, xAI (2025). URL https://data.x.ai/2025-08-20-grok-4-model-card.pdf
74. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., Rush, A.M.: Transformers: State-of-the-art natural language processing. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-demos.6
75. Luo, X., Zhang, Y., He, Z., Wang, Z., Zhao, S., Li, D., Qiu, L.K., Yang, Y.: Agent Lightning: Train ANY AI agents with reinforcement learning. arXiv preprint arXiv:2508.03680 (2025). URL https://arxiv.org/abs/2508.03680
76. Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.H., Choo, J.: Reversible instance normalization for accurate time-series forecasting against distribution shift. In: International Conference on Learning Representations (2022). URL https://openreview.net/forum?id=cGDAkQo1C0p
77. Meta: Llama 3.2 model card. Model card (2024). URL https://huggingface.co/meta-llama/Llama-3.2-3B. Accessed: Oct. 15, 2026
78. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Renard Lavaud, L., Lachaux, M.A., Stock, P., Le Scao, T., Lavril, T., Wang, T., Lacroix, T., El Sayed, W.: Mistral 7B. arXiv preprint arXiv:2310.06825 (2023). URL https://arxiv.org/abs/2310.06825