PAST ERRORS, BETTER FORECASTS: MEMORY-GUIDED CALIBRATION FOR FROZEN TIME SERIES PREDICTORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series forecasting supports decision making in many data-intensive applications, where forecasting models typically use a fixed-length historical window to predict a future sequence. In standard forecasting pipelines, a single checkpoint is typically selected according to aggregate validation performance, yet strong aggregate performance does not guarantee uniformly small errors across samples, variables, or temporal contexts. Consequently, a frozen forecaster may exhibit structured, input-dependent residual patterns on historical labeled windows, which can provide reusable evidence of how the same forecaster tends to deviate under similar temporal conditions. Motivated by this observation, we propose Memory-Guided Test-Time Calibration (TTC), a non-parametric residual-memory framework that calibrates frozen time-series forecasters at the prediction level. After model selection, TTC constructs an external residual memory that associates compact variable-wise representations of historical look-back windows with their corresponding true-minus-predicted residual trajectories. At inference time, the current look-back window is used to retrieve historically similar variable patterns, and their residual trajectories are similarity-weighted and aggregated to compensate the base prediction. To reduce harmful over-correction, TTC applies a correction-ratio fallback that preserves the base prediction when the relative magnitude of the candidate correction is excessively large. The forecasting backbone remains frozen throughout calibration, and TTC never accesses the future labels of the current test window. Experiments across eight benchmark datasets and five heterogeneous forecasting backbones, together with comparisons against representative test-time adaptation methods and extensive ablation and sensitivity analyses, demonstrate the effectiveness and robustness of TTC across diverse forecasting settings.

1 INTRODUCTION

Time series forecasting is a fundamental task in many data-intensive domains, including industrial, energy, financial, meteorological, and public-health applications, where accurate forecasts support downstream decision making Lim & Zohren (2021); Petropoulos et al. (2022). Modern forecasting models typically take a fixed-length historical window as input and predict a future sequence over a predefined horizon. These models are commonly trained offline, and a single checkpoint is selected according to aggregate validation performance, typically with early stopping. Such model selection aims to identify a checkpoint that performs well on the validation set as a whole, but does not guarantee uniformly strong performance across individual samples, variables, or temporal contexts.

Consequently, even a validation-selected forecaster with strong aggregate performance may exhibit heterogeneous residual patterns across samples, variables, and temporal contexts. Importantly, once the forecasting checkpoint has been fixed, these residuals are directly observable on historical labeled windows, for which both the model prediction and the corresponding future observations are already available. The remaining errors need not be purely random prediction noise; under some temporal conditions, they may exhibit systematic and input-dependent structure. Under similar historical patterns, a frozen forecaster may repeatedly exhibit related forms of deviation, such as underestimating peaks, over-smoothing abrupt changes, or lagging behind local temporal variations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

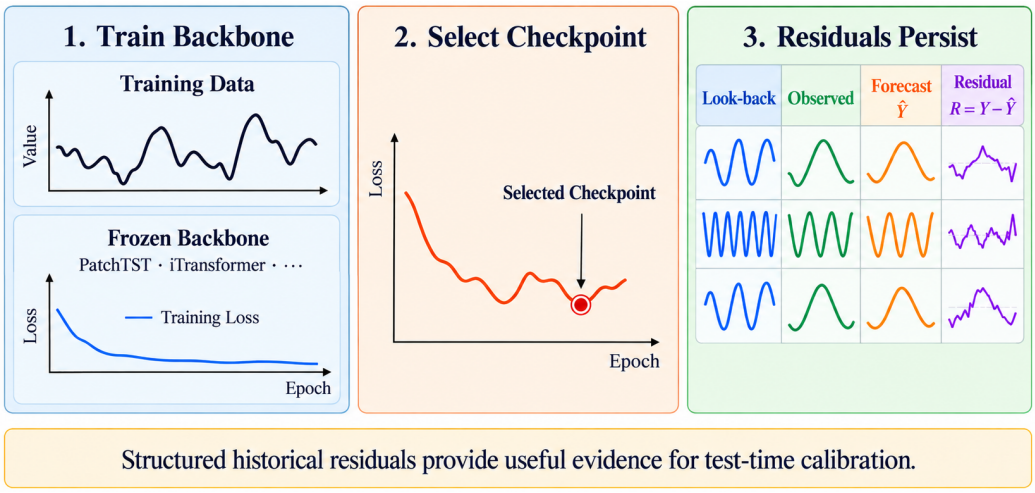


Figure 1: Motivation for historical residual memory. (1) A forecasting backbone is trained on historical data. (2) A checkpoint is selected according to aggregate validation performance and then kept frozen. (3) Historical labeled windows reveal residual trajectories produced by the frozen forecaster, which can be stored as external evidence for prediction-level calibration.

Historical residuals may therefore provide reusable evidence about how the same frozen forecaster tends to deviate under different input conditions. This observation motivates us to store residual information from previously observed windows and reuse it as external evidence for prediction-level calibration. Figure 1 summarizes this motivation.

Turning such historical residuals into reliable calibration evidence, however, is non-trivial. First, historical errors cannot simply be treated as global offsets, because the residual behavior of a frozen forecaster may depend on the current sample, variable, and temporal context. A calibration mechanism must therefore identify which historical patterns are relevant to the current input and whether their associated residual trajectories provide a useful correction signal. Second, defining similarity is challenging in multivariate forecasting: different variables may exhibit different scales, local trends, periodic phases, and fluctuation patterns, while a single global similarity measure can obscure variable-specific behavior. A useful retrieval representation should also remain applicable across heterogeneous forecasting backbones rather than relying on model-specific hidden states. Third, even historically similar patterns may provide noisy or mismatched residual evidence. Applying retrieved residuals unconditionally can therefore over-correct an otherwise reliable base prediction. A practical residual-calibration mechanism must jointly address context-dependent retrieval, variable-wise residual estimation, and conservative fallback.

To address these challenges, we propose Memory-Guided Test-Time Calibration (TTC), a model-agnostic residual-memory framework for prediction-level calibration of frozen time-series forecasters. After a forecasting backbone has been trained and selected, TTC constructs an external residual memory bank from historical labeled windows. Each memory entry associates a compact variable-wise representation of a historical look-back window with the corresponding true-minus-predicted residual trajectory produced by the frozen forecaster. At inference time, TTC encodes the current look-back window in the same retrieval space, identifies historically similar patterns for each variable, and aggregates multiple retrieved residual trajectories to form a local correction estimate. To reduce harmful over-correction, TTC applies a correction-ratio fallback that rejects candidate corrections whose magnitude is excessively large relative to the base prediction and preserves the original frozen-backbone output in such cases. The entire calibration process operates without updating the forecasting backbone or accessing the future labels of the current test window.

The main contributions of this work are summarized as follows:

- We formulate prediction-level calibration of frozen time-series forecasters as conditional residual estimation and instantiate this view with Memory-Guided Test-Time Calibration (TTC), a non-parametric residual-memory framework that operates without test-time parameter updates.

- We design a compact variable-wise retrieval and residual estimation mechanism that combines statistical properties, normalized temporal shape, and first-order differences. TTC retrieves multiple historically similar patterns for each variable and aggregates their associated residual trajectories to form a local correction estimate, avoiding dependence on a single historical neighbor or backbone-specific hidden states.
- We introduce a correction-ratio fallback that suppresses overly aggressive residual compensation by preserving the base prediction when the relative correction magnitude becomes excessive. We evaluate TTC across eight benchmark datasets, five heterogeneous forecasting backbones, and multiple prediction horizons, including comparisons with representative test-time adaptation methods and controlled ablation and sensitivity analyses.

2 Related Work

Test-time adaptation (TTA) aims to improve the deployment-time performance of a pretrained model by exploiting information available during inference, particularly when the test distribution differs from the data observed during training (Wang et al., 2021; 2022; Niu et al., 2022; 2023; Zhang et al., 2022a). Existing TTA methods adapt components such as normalization statistics, selected model parameters, prediction layers, or lightweight auxiliary modules using signals from unlabeled test inputs, self-supervised objectives, or newly observed data. Extending this paradigm to time-series forecasting is particularly challenging because the future target associated with the current prediction is inherently unavailable when the forecast is issued. Time-series TTA methods therefore often exploit the sequential nature of forecasting and use information that becomes available during deployment, including newly observed values, delayed feedback, partially observed future targets, or contextual signals, to adapt the forecasting process.

Several recent studies have explored test-time adaptation and context-aware calibration for time-series forecasting. TAFAS addresses forecasting under non-stationarity through periodicity-aware adaptation and a gated calibration mechanism, exploiting partially observed test-time ground truth as the sequence evolves (Kim et al., 2025). SOLID considers sample-level contextualized adaptation, where historical samples that are contextually similar to the current test sample are selected to construct a local adaptation set, which is subsequently used to fine-tune the prediction layer for the current context (Chen et al., 2024). These approaches demonstrate that information available after offline model training can be useful for improving deployment-time forecasting, but they differ substantially in the information they assume to be available and in how they modify the forecasting process. In particular, TAFAS relies on online adaptation with newly observed test-time information, while SOLID performs gradient-based adaptation using contextually selected historical samples.

Similarity-based and retrieval-inspired techniques have also been explored in time-series representation learning, forecasting, and contextualized adaptation (Yue et al., 2022; Woo et al., 2022; Tonekaboni et al., 2021; Li et al., 2023; Dong et al., 2023; Cheng et al., 2026). These approaches broadly exploit relationships among temporally related historical patterns, but input similarity alone does not imply identical future behavior or identical forecasting error. TTC builds on this observation from a different perspective: rather than retrieving historical samples to directly predict the future or to update model parameters, it retrieves residual trajectories produced by the same frozen forecaster under similar historical conditions. Multiple variable-wise residual trajectories are then aggregated to form a non-parametric local correction, while a conservative correction-ratio fallback suppresses overly large adjustments. TTC therefore operates entirely at the prediction level, without modifying the forecasting backbone or accessing the future labels of the current test window. A broader discussion of forecasting architectures, general test-time adaptation methods, and similarity-based time-series approaches is provided in the appendix.

3 Method

Figure 2 illustrates Memory-Guided Test-Time Calibration (TTC), which consists of offline residual-memory construction and online test-time residual calibration. The central idea is to reuse residual trajectories observed on historical labeled windows as external correction signals for future predictions. During the offline stage, TTC applies the frozen forecaster to historical labeled windows and stores their variable-wise retrieval representations together with the corresponding residual trajec-

tories. At test time, the current look-back window produces both the base forecast and a query representation; TTC retrieves similar historical patterns, aggregates their residual trajectories into a candidate correction, and applies a conservative correction-ratio gate before updating the prediction. The forecasting backbone remains frozen throughout calibration, and the future labels of the current test window are never accessed.

3.1 Problem Setup and Framework Overview

Let $\mathbf{X}_t \in \mathbb{R}^{L\times C}$ denote a multivariate look-back window, where L is the look-back length and C is the number of variables. Let $\mathbf{Y}_t \in \mathbb{R}^{H\times C}$ denote the corresponding future window with prediction horizon H. A forecasting backbone f_θ maps the observed history to a base prediction:

$$\hat{\mathbf{Y}}_t = f_\theta(\mathbf{X}_t). \tag{1}$$

The backbone is trained using its original forecasting objective, and a checkpoint is selected according to validation performance. Once the checkpoint is fixed, TTC does not alter its architecture, training objective, or parameters.

Given the frozen backbone, we view prediction-level calibration as conditional residual estimation. For variable c, let $\mathbf{R}_c$ denote its H-step true-minus-predicted residual trajectory and $\mathbf{Z}_c$ an observable representation of its look-back context; the population target is $\mathbf{b}_c^*(\mathbf{z}) = \mathbb{E}[\mathbf{R}_c \mid \mathbf{Z}_c = \mathbf{z}]$. TTC approximates this context-dependent residual non-parametrically from historical residual memory, while the current test window is used only to query the fixed memory. A detailed statistical interpretation is provided in Appendix B.1.

3.2 Residual Memory Bank Construction

Let $\mathcal{D}_h = \{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^N$ denote the historical labeled windows used for memory construction. These samples correspond to past windows whose future values have already been observed, rather than to the unknown future of the current test window. For each historical window, the frozen forecaster produces

$$\hat{\mathbf{Y}}_i = f_\theta(\mathbf{X}_i), \qquad \mathbf{R}_i = \mathbf{Y}_i - \hat{\mathbf{Y}}_i, \tag{2}$$

where $\mathbf{R}_i \in \mathbb{R}^{H\times C}$ records the true-minus-predicted residual trajectories over the prediction horizon.

A residual trajectory alone does not indicate the historical input condition under which it was produced. TTC therefore associates each residual with a retrieval representation of its historical look-back window. Rather than constructing a single representation for the entire multivariate window, TTC defines a separate representation for each variable:

$$\mathbf{e}_{i,c} = \frac{\phi_c(\mathbf{X}_i)}{\|\phi_c(\mathbf{X}_i)\|_2 + \epsilon}, \tag{3}$$

where $\phi_c(\cdot)$ denotes the retrieval representation for variable c and ϵ is a small constant for numerical stability. Let

$$\mathbf{E}_i = [\mathbf{e}_{i,1}; \ldots; \mathbf{e}_{i,C}], \tag{4}$$

where $[\,;\,]$ denotes stacking of the variable-wise representations. The residual memory bank is then defined as

$$\mathcal{M} = \{(\mathbf{E}_i, \mathbf{R}_i)\}_{i=1}^N. \tag{5}$$

Each memory item therefore associates a representation of the historical input condition with the residual trajectory produced by the same frozen forecaster. The memory bank is constructed before final test-time calibration and remains fixed during evaluation.

3.3 Variable-Wise Similarity Retrieval

A key question is how to determine whether a historical pattern is relevant to the current input. Direct comparison of raw values preserves absolute level information but is sensitive to scale, offset, and variable-specific magnitude. In contrast, using only normalized temporal shape reduces these effects but can discard useful information about local level and variability. Furthermore, a single

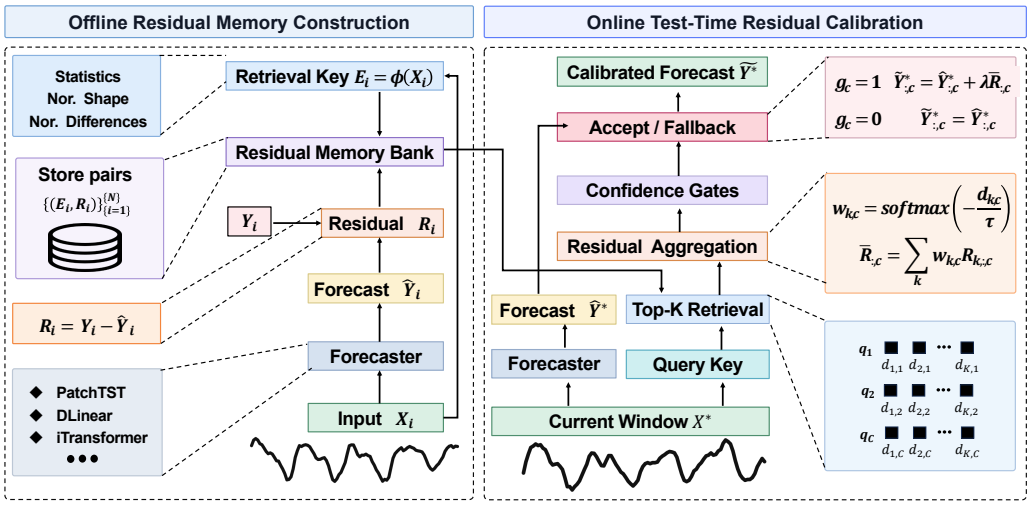


Figure 2: Overview of TTC. Offline, historical labeled windows are used to construct a residual memory bank that stores variable-wise retrieval representations together with their corresponding residual trajectories. At test time, the current window is used to retrieve and aggregate relevant historical residuals, after which a conservative correction-ratio gate determines whether to apply the candidate correction or fall back to the base forecast. The forecasting backbone remains frozen throughout the procedure.

similarity representation for the entire multivariate window may obscure variable-specific temporal behavior.

To balance these factors, TTC defines similarity independently for each variable using a compact representation that combines three complementary feature blocks. The statistical block summarizes local level and variability, the normalized-shape block characterizes the standardized temporal trajectory, and the difference block captures normalized first-order temporal changes. For variable c,

$$\phi_c(\mathbf{X}) = \mathrm{Concat}_{m\in\mathcal{B}}\left(\alpha_m \phi_{m,c}(\mathbf{X})\right), \qquad \mathcal{B} = \{\mathrm{stat}, \mathrm{shape}, \mathrm{diff}\}, \tag{6}$$

where α_m controls the contribution of feature block m. The representation is computed directly from the observed time series and does not depend on backbone-specific hidden states, allowing the same retrieval procedure to be applied across heterogeneous forecasting models.

For a current look-back window $\mathbf{X}_t$, the corresponding variable-wise query is

$$\mathbf{q}_{t,c} = \frac{\phi_c(\mathbf{X}_t)}{\|\phi_c(\mathbf{X}_t)\|_2 + \epsilon}. \tag{7}$$

TTC compares the query with the stored historical representation of the same variable using squared Euclidean distance:

$$d_{i,c} = \|\mathbf{q}_{t,c} - \mathbf{e}_{i,c}\|_2^2. \tag{8}$$

The K historical items with the smallest distances form the neighbor set $\mathcal{N}_K^{(c)}(t)$.

Variable-wise retrieval allows different variables in the same test window to select different historical neighbors. The resulting neighbor sets therefore capture historical conditions that are locally relevant to each variable rather than enforcing one globally similar multivariate window.

3.4 Residual Compensation and Conservative Fallback

After retrieving the top-K historical patterns, TTC uses their associated residual trajectories to estimate a correction for the current base prediction. Closer historical patterns receive larger weights through a temperature-scaled softmax:

$$w_{i,c} = \frac{\exp(-d_{i,c}/\tau)}{\sum_{j\in\mathcal{N}_K^{(c)}(t)} \exp(-d_{j,c}/\tau)}, \qquad i \in \mathcal{N}_K^{(c)}(t), \tag{9}$$

where τ controls the concentration of the weighting distribution over the retrieved neighbors.

The residual correction for variable c is estimated as

$$\hat{\mathbf{b}}_{t,c} = \sum_{i \in \mathcal{N}_K^{(c)}(t)} w_{i,c} \mathbf{R}_{i,:,c}. \tag{10}$$

This aggregation can be viewed as a finite-memory local estimate of the conditional residual introduced in Section 3.1; its population interpretation and relation to residual shrinkage are detailed in Appendix B.1. The resulting candidate prediction is

$$\tilde{\mathbf{Y}}_{t,:,c} = \hat{\mathbf{Y}}_{t,:,c} + \lambda \hat{\mathbf{b}}_{t,c}, \tag{11}$$

where λ controls the strength of residual compensation.

Input similarity does not guarantee that a historical residual correction is fully transferable to the current test window. TTC therefore applies a variable-wise correction-ratio fallback. For variable c, the relative magnitude of the candidate correction is

$$\rho_{t,c} = \frac{\|\lambda \hat{\mathbf{b}}_{t,c}\|_2}{\|\hat{\mathbf{Y}}_{t,:,c}\|_2 + \epsilon}. \tag{12}$$

The correction is accepted only when

$$a_{t,c} = \mathbb{I}\left[\rho_{t,c} \leq \delta_r\right], \tag{13}$$

where δ_r is a predefined correction-ratio threshold. The final prediction is

$$\mathbf{Y}^{\text{final}}_{t,:,c} = a_{t,c} \tilde{\mathbf{Y}}_{t,:,c} + (1 - a_{t,c}) \hat{\mathbf{Y}}_{t,:,c}. \tag{14}$$

Thus, TTC applies the retrieved residual correction only when its magnitude remains within the predefined range relative to the base prediction; otherwise, it falls back to the original frozen-backbone output. The complete test-time procedure uses only the observed look-back window, the frozen-backbone prediction, and the pre-built residual memory, requiring neither gradient-based parameter updates nor access to the future labels of the current test window.

4 Experiments

We evaluate TTC from three complementary perspectives: overall forecasting performance across heterogeneous forecasting settings, comparison with representative test-time adaptation methods under a shared-base protocol, and controlled ablation and sensitivity analyses of the main calibration components.

4.1 Experimental Setup

Datasets and forecasting backbones. We evaluate TTC on eight multivariate time-series forecasting benchmarks: ETTh1, ETTh2, ETTm1, ETTm2, Exchange, Weather, Illness, and Solar. For all datasets except Illness, we use prediction horizons $H \in \{96, 192, 336, 720\}$; Illness uses $H \in \{24, 36, 48, 60\}$. We consider five representative forecasting backbones: iTransformer (Liu et al., 2023a), PatchTST (Nie et al., 2023), DLinear (Zeng et al., 2023), Autoformer (Wu et al., 2021), and Crossformer (Zhang & Yan, 2023). These models cover variable-centric Transformer, patch-based, linear, decomposition-based, and cross-dimensional forecasting architectures.

Training and TTC protocol. Each forecasting backbone is trained using its original forecasting objective, and the checkpoint is selected according to validation performance. The selected checkpoint is then frozen throughout calibration and final test evaluation. TTC operates only at the prediction level and does not update the forecasting backbone. The residual memory is constructed from historical windows whose corresponding future observations are already available. The future labels of the current test window are never used for similarity retrieval, residual aggregation, fallback decisions, or final prediction.

Comparison baselines and shared-base protocol. We compare TTC with TAFAS (Kim et al., 2025) and SOLID (Chen et al., 2024). To avoid confounding test-time adaptation performance

Table 1: Horizon-averaged comparison of TAFAS, SOLID, and TTC under the shared-base protocol. MSE and MAE are averaged over $H \in \{96, 192, 336, 720\}$, except for Illness with $H \in \{24, 36, 48, 60\}$. Lower values are better. Boldface indicates the best result according to the unrounded values.

Model	Dataset	Base		TAFAS		SOLID		TTC (Ours)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
iTransformer	ETTh1	0.453	0.447	0.451	0.450	0.446	0.444	**0.441**	**0.443**
	ETTh2	0.384	0.407	0.385	0.408	0.384	0.407	**0.370**	**0.397**
	ETTm1	0.408	0.412	0.393	0.410	0.408	0.412	**0.384**	**0.402**
	ETTm2	0.291	0.335	0.294	0.338	0.291	**0.335**	**0.284**	0.335
	Weather	0.260	0.280	0.245	0.278	0.260	0.280	**0.238**	**0.275**
	Exchange	0.365	0.408	0.348	0.399	0.365	0.408	**0.264**	**0.374**
	Illness	2.337	1.037	2.335	1.036	2.369	1.030	**1.907**	**0.888**
	Solar	0.235	0.262	0.234	0.267	0.234	0.262	**0.218**	**0.255**
PatchTST	ETTh1	0.423	0.434	0.424	0.436	0.418	**0.430**	**0.418**	0.432
	ETTh2	0.345	0.387	0.345	0.387	0.345	0.387	**0.337**	**0.383**
	ETTm1	0.353	0.383	0.353	0.384	0.353	0.382	**0.346**	**0.378**
	ETTm2	0.258	0.315	0.256	0.316	0.258	0.315	**0.252**	**0.311**
	Weather	0.229	0.264	0.228	0.271	0.229	0.264	**0.221**	**0.260**
	Exchange	0.385	0.420	0.369	0.413	0.385	0.420	**0.303**	**0.388**
	Illness	1.937	0.923	1.934	0.921	2.005	0.914	**1.720**	**0.840**
	Solar	0.246	0.277	0.250	0.285	0.245	0.276	**0.229**	**0.269**
DLinear	ETTh1	0.422	0.436	0.420	**0.431**	**0.417**	0.432	0.418	0.434
	ETTh2	0.425	0.442	0.385	0.419	0.412	0.434	**0.370**	**0.407**
	ETTm1	0.358	0.381	0.354	0.378	0.358	0.381	**0.341**	**0.375**
	ETTm2	0.277	0.341	0.264	0.324	0.276	0.341	**0.256**	**0.322**
	Weather	0.248	0.304	0.240	0.288	0.248	0.303	**0.228**	**0.277**
	Exchange	0.332	0.398	0.310	0.388	0.326	0.394	**0.297**	**0.380**
	Illness	2.255	1.062	2.245	1.059	2.235	1.040	**1.918**	**0.963**
	Solar	0.326	0.397	0.322	0.396	0.326	0.397	**0.264**	**0.337**
Autoformer	ETTh1	0.488	0.483	0.487	0.484	1.425	0.884	**0.471**	**0.473**
	ETTh2	0.436	0.452	0.442	0.455	0.443	0.455	**0.413**	**0.439**
	ETTm1	0.533	0.496	0.508	0.484	0.532	0.496	**0.471**	**0.465**
	ETTm2	0.337	0.372	0.330	0.371	0.337	0.372	**0.322**	**0.366**
	Weather	0.344	0.386	0.310	0.361	0.344	0.386	**0.306**	**0.359**
	Exchange	0.631	0.568	0.617	0.561	0.631	0.568	**0.359**	**0.443**
	Illness	3.286	1.249	3.282	1.248	3.275	1.246	**2.722**	**1.107**
	Solar	0.693	0.617	0.537	0.549	0.697	0.618	**0.377**	**0.438**
Crossformer	ETTh1	0.465	0.480	**0.427**	**0.450**	0.545	0.539	0.440	0.465
	ETTh2	1.200	0.787	0.984	0.709	0.954	0.694	**0.744**	**0.606**
	ETTm1	0.396	0.419	0.369	0.404	0.393	0.417	**0.368**	**0.402**
	ETTm2	0.579	0.514	**0.334**	**0.388**	0.570	0.510	0.371	0.396
	Weather	0.229	0.289	0.225	0.283	0.228	0.289	**0.218**	**0.274**
	Exchange	0.791	0.679	0.759	0.661	0.754	0.665	**0.632**	**0.596**
	Illness	3.460	1.203	3.457	1.202	**3.135**	**1.158**	3.436	1.205
	Solar	0.336	0.346	0.335	0.349	0.336	0.346	**0.230**	**0.271**

with differences in source-model quality, all compared methods start from the same validation-selected forecasting checkpoint for each model–dataset–horizon setting and are evaluated on the same forecasting data. This shared-base protocol allows the comparison to focus on the additional effect introduced by each test-time adaptation or calibration mechanism. Further implementation details, baseline configuration coverage, and TTC calibration settings are provided in the appendix.

Evaluation metrics. We report mean squared error (MSE) and mean absolute error (MAE), where lower values indicate better forecasting accuracy. Relative improvements are reported with respect to the corresponding shared Base prediction.

Table 2: Ablation study of residual calibration strategies on Weather with $H = 96$. TTC combines similarity-weighted Top-K residual aggregation with the correction-ratio fallback. Lower MSE and MAE are better; boldface indicates the best result according to the unrounded values.

Backbone	Metric	Base	Mean	Random-K	1-NN	Top-K-U	Top-K-W	TTC
iTransformer	MSE	0.175	0.175	0.178	0.258	0.164	0.165	**0.163**
	MAE	0.214	0.219	0.225	0.249	0.208	0.209	**0.206**
PatchTST	MSE	0.150	0.149	0.151	0.268	0.145	0.146	**0.145**
	MAE	0.197	0.200	0.205	0.246	0.197	0.198	**0.195**
DLinear	MSE	0.178	0.178	0.182	0.349	0.157	0.161	**0.156**
	MAE	0.243	0.244	0.252	0.286	0.218	0.218	**0.214**
Autoformer	MSE	0.269	0.265	0.268	0.441	0.251	0.251	**0.251**
	MAE	0.343	0.338	0.342	0.418	0.328	0.328	**0.327**
Crossformer	MSE	0.148	0.145	0.148	0.243	0.143	0.147	**0.143**
	MAE	0.216	0.211	0.214	0.244	0.207	0.207	**0.205**

4.2 Main Results

We evaluate TTC across five forecasting backbones, eight datasets, and four prediction horizons, yielding 160 model–dataset–horizon settings. Complete horizon-wise Base, TAFAS, SOLID, and TTC results are reported by forecasting backbone in Appendix C. Overall, TTC provides improvements across Transformer-based, patch-based, linear, decomposition-based, and cross-dimensional forecasting architectures, although the magnitude of the gain varies across datasets and prediction horizons.

Under the shared-base protocol described above, Table 1 reports horizon-averaged MSE and MAE for each backbone–dataset pair. Results are averaged over $H \in \{96, 192, 336, 720\}$ for all datasets except Illness, which uses $H \in \{24, 36, 48, 60\}$.

TTC achieves the lowest horizon-averaged MSE in 36 of the 40 backbone–dataset combinations and the lowest MAE in 34 combinations. The gains span all five forecasting architectures and are particularly substantial in several model–dataset pairs, while remaining more modest in others. This variation is consistent with the residual-calibration setting, where the usefulness of historical residual correction depends on whether relevant residual patterns are available for the current forecasting context.

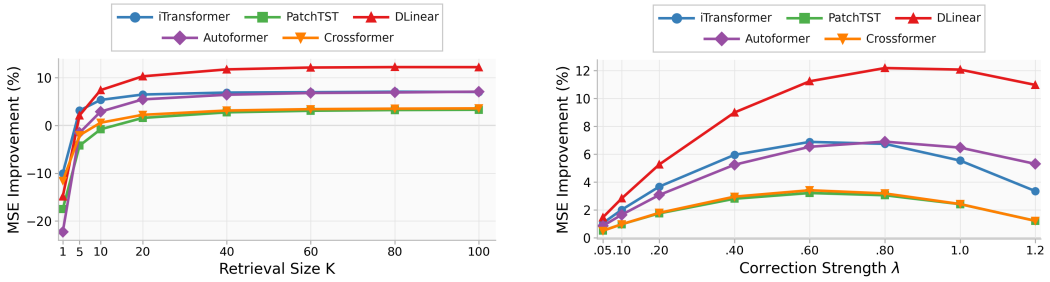
TTC does not dominate every setting: TAFAS performs best in some Crossformer cases, while SOLID achieves the best result for a small number of backbone–dataset pairs. These exceptions suggest that the effectiveness of different test-time mechanisms can vary with the forecasting architecture and data characteristics. Overall, the results show that TTC provides a competitive prediction-level calibration mechanism across heterogeneous forecasting settings while keeping the forecasting backbone fixed.

4.3 Ablation and Sensitivity Analysis

We examine the key components of TTC and its sensitivity to the main calibration hyperparameters, focusing in the main text on the residual-calibration strategy, retrieval size K, and correction strength λ. Additional analyses of the retrieval representation, correction-ratio decision statistics, softmax temperature, memory stride, and horizon-wise parameter sensitivity are provided in Appendix D.

Residual calibration strategy. Table 2 compares non-contextual residual reuse, single-neighbor retrieval, multi-neighbor aggregation, similarity weighting, and the complete TTC framework on Weather with $H = 96$ across five forecasting backbones. For each strategy, we report the frozen Base performance, the calibrated result, and the corresponding relative MSE and MAE improvements.

The results show a clear difference between context-independent residual reuse and retrieval-based calibration. Mean residual correction provides only limited improvements, while Random-K fre-



(a) Retrieval size K

(b) Correction strength λ

Figure 3: Cross-backbone sensitivity of TTC on Weather with $H = 96$. Panels report MSE improvement over the frozen Base as (a) retrieval size K and (b) correction strength λ vary.

quently degrades the Base prediction, indicating that historical residuals should not be treated as input-independent additive offsets. Although input-dependent retrieval is important, 1-NN is highly unstable because the correction depends entirely on a single historical residual trajectory; in contrast, Top-K-U and Top-K-W produce substantially more stable improvements across the five backbones.

Both uniform and similarity-weighted Top-K aggregation provide useful residual estimates, although similarity weighting alone does not consistently outperform uniform aggregation. The complete TTC framework further applies the correction-ratio fallback and achieves the best calibrated MSE and MAE for all five backbones. The fallback rejects only approximately 1.2%–8.0% of variable-wise corrections, suggesting that it mainly filters a small number of over-amplified corrections. Detailed decision statistics are reported in Appendix D.1.

Sensitivity to K and λ. The retrieval size K determines how many historical residual trajectories are used to estimate the correction, while λ controls the magnitude of the residual compensation applied to the Base prediction.

Figure 3(a) shows that very small K values can be unstable and may even degrade the Base prediction. As more neighbors are aggregated, the MSE improvement rises rapidly and then approaches a broad plateau, supporting multi-neighbor aggregation and indicating that TTC is not tied to a narrowly tuned retrieval size.

Figure 3(b) shows that very small λ values underuse the retrieved residual information, while increasing λ improves performance over a broad intermediate-to-high range. Beyond this region, the gain saturates or decreases for several backbones, consistent with increasingly aggressive residual compensation. Overall, the sensitivity results indicate a broad stable operating region rather than dependence on a single exact value of either K or λ.

Additional sensitivity results, including horizon-wise analyses of K and λ, softmax temperature τ, and memory stride, are reported in Appendix D.

5 Conclusion

In this paper, we presented Memory-Guided Test-Time Calibration (TTC), a non-parametric prediction-level calibration framework for frozen time-series forecasters. TTC reuses historical residual information from labeled past windows through variable-wise residual retrieval and conservative correction, while keeping the forecasting backbone frozen and never accessing the future labels of the current test window. Extensive experiments across eight benchmark datasets, five representative forecasting backbones, and multiple prediction horizons show that historical residual memory can improve frozen forecasters across diverse forecasting settings. Comparisons with representative test-time adaptation methods, together with ablation and sensitivity analyses, further support the effectiveness of variable-wise residual retrieval, multi-neighbor aggregation, and conservative correction. Overall, TTC offers a model-agnostic prediction-level calibration mechanism that complements frozen forecasting backbones through external historical residual memory.

AI USE STATEMENT

Generative AI tools were used solely to assist with language polishing, grammatical refinement, and improving the clarity and readability of the manuscript. All research ideas, methodological design, experimental procedures, results, analyses, and scientific conclusions were developed and verified by the authors. No AI tools were used to generate experimental results or make scientific decisions.

REPRODUCIBILITY STATEMENT

We provide the methodological and experimental details necessary to support reproducibility throughout the main paper and appendices. These include the forecasting and calibration protocols, residual-memory construction, retrieval and correction procedures, shared-base evaluation protocol, TTC configurations, baseline configuration coverage, complete horizon-wise results, and additional ablation and sensitivity analyses.

REFERENCES

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Mouxiang Chen, Lefei Shen, Han Fu, Zhuo Li, Jianling Sun, and Chenghao Liu. Calibration of time-series forecasting: Detecting and adapting context-driven distribution shift. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 341–352, 2024.

Si Chen, Chun-Liang Li, Nate Yoder, Sercan O. Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023.

Mingyue Cheng, Xiaoyu Tao, Zhiding Liu, Qi Liu, Hao Zhang, Rujiao Zhang, and Enhong Chen. TimeMAE: Self-supervised representations of time series with decoupled masked autoencoders. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, WSDM '26, pp. 498–508. Association for Computing Machinery, 2026. doi: 10.1145/3773966.3778007.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023.

Jiaxiang Dong, Haixu Wu, Haoran Zhang, Li Zhang, Jianmin Wang, and Mingsheng Long. Simmtm: A simple pre-training framework for masked time-series modeling. In *Advances in Neural Information Processing Systems*, 2023.

Wei Fan, Pengyang Wang, Dongkun Wang, Dongjie Wang, Yuanchun Zhou, and Yanjie Fu. Dish-ts: A general paradigm for alleviating distribution shift in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7522–7529, 2023.

HyunGi Kim, Siwon Kim, Jisoo Mok, and Sungroh Yoon. Battling the non-stationarity in time series forecasting via test-time adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 17868–17876, 2025.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in Neural Information Processing Systems*, 2019.

Zhe Li, Zhongwen Rao, Lujia Pan, Pengyun Wang, and Zenglin Xu. Ti-mae: Self-supervised masked time series autoencoders. *arXiv preprint arXiv:2301.08871*, 2023.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical transactions of the royal society a: mathematical, physical and engineering sciences*, 379(2194), 2021.

Bryan Lim, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 2021.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in neural information processing systems*, 35: 9881–9893, 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023a.

Zhiding Liu, Mingyue Cheng, Zhi Li, Zhenya Huang, Qi Liu, Yanhu Xie, and Enhong Chen. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b.

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

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. *arXiv preprint arXiv:2302.12400*, 2023.

Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J Bessa, Jakub Bijak, John E Boylan, et al. Forecasting: theory and practice. *International Journal of forecasting*, 38(3):705–871, 2022.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2020.

Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. *arXiv preprint arXiv:2106.00750*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uXl3bZLkr3c.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7201–7211, 2022.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=PilZY3omXV2.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 8980–8987, 2022.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.

Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in neural information processing systems*, 35:38629–38642, 2022a.

Tian Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022b.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pp. 11106–11115. AAAI Press, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proc. 39th International Conference on Machine Learning (ICML 2022)*, 2022.

A Extended Background and Related Work

This appendix provides additional background on time-series forecasting architectures, context-dependent forecasting residuals, test-time adaptation, and similarity-based time-series methods. The discussion complements the concise related-work section in the main paper and provides broader context for prediction-level residual calibration.

A.1 Forecasting Background and Representative Backbones

Time-series forecasting has been studied with a wide range of neural architectures designed to capture temporal dependencies, periodic patterns, and cross-variable relationships. Earlier neural approaches include recurrent, convolutional, probabilistic, and block-based forecasting models (Lai et al., 2018; Salinas et al., 2020; Bai et al., 2018; Oreshkin et al., 2019; Lim et al., 2021). Following the success of attention-based sequence modeling (Vaswani et al., 2017), Transformer-style architectures became increasingly common in long-term forecasting. LogTrans introduces locality-aware attention for long sequences, Informer reduces attention complexity through sparse attention, Autoformer combines decomposition with an auto-correlation mechanism, and FEDformer incorporates frequency-domain representations into the forecasting process (Li et al., 2019; Zhou et al., 2021; Wu et al., 2021; Zhou et al., 2022).

More recent forecasting architectures explore diverse representations of temporal and cross-variable structure. PatchTST partitions each variable into temporal patches and performs channel-independent modeling (Nie et al., 2023); Crossformer explicitly models cross-time and cross-dimension dependencies (Zhang & Yan, 2023); and iTransformer treats variables as tokens to strengthen multivariate dependency modeling (Liu et al., 2023a). Lightweight alternatives have also been widely explored. DLinear uses decomposition-based linear forecasting, while LightTS, TSMixer, TiDE, and TimeMixer investigate lightweight sampling, MLP-based mixing, dense encoding, and multiscale temporal modeling (Zeng et al., 2023; Zhang et al., 2022b; Chen et al., 2023; Das et al., 2023; Wang et al., 2024).

A.2 Historical Residuals and Context-Dependent Forecasting Errors

Most forecasting models are trained to optimize an aggregate objective over a collection of training samples, and the final checkpoint is typically selected according to aggregate validation performance. This procedure is designed to identify a model that performs well on average, but the resulting objective does not require forecasting errors to be equally small for every individual sample, variable, or temporal context.

Consequently, a validation-selected forecaster may achieve strong overall performance while still exhibiting heterogeneous residual behavior across historical windows. Some samples may already be predicted accurately, whereas others may retain larger or more structured errors. Similarly, different variables within the same multivariate series can exhibit different residual characteristics because they may differ in scale, periodicity, local trend, volatility, and predictability. These differences can also vary with the forecasting horizon.

Once the forecasting checkpoint has been fixed, such residual behavior can be examined on historical labeled windows for which both the model prediction and the corresponding future observations are already available. For a historical sample, the residual records not only the magnitude of the forecasting error but also its temporal trajectory across the prediction horizon. Under particular input conditions, a frozen forecaster may repeatedly exhibit related forms of deviation, such as underestimating peaks, over-smoothing abrupt changes, lagging behind changing periodic patterns, or extrapolating a local trend for too long.

Importantly, this does not imply that forecasting residuals are universally predictable or that similar inputs must produce identical errors. A substantial portion of the remaining forecasting error may still be irregular or difficult to reuse. The relevant observation is more limited: aggregate model selection can leave locally structured residual behavior for some samples, variables, and temporal contexts, and these historical residuals are directly observable after their corresponding outcomes have occurred.

This distinction is important for prediction-level calibration. If the remaining error were purely unstructured, historical residual reuse would provide little useful information beyond the Base forecast. Conversely, when related historical contexts exhibit partially reusable error patterns, those residuals can provide additional evidence about how the same frozen forecaster has behaved under similar conditions. The usefulness of such evidence therefore depends jointly on the residual structure of the Base forecaster and on whether sufficiently relevant historical contexts are available.

Context-dependent residual behavior is related to, but distinct from, general non-stationarity and distribution-shift modeling. Existing normalization, decomposition, and stationarization approaches aim to improve forecasting robustness by modifying the input representation or forecasting process under changing temporal statistics (Kim et al., 2021; Liu et al., 2022; Fan et al., 2023; Liu et al., 2023b). These approaches can improve the Base forecaster itself, but they do not necessarily eliminate all sample-, variable-, or context-dependent residuals after training and checkpoint selection.

Recent context-aware forecasting work similarly suggests that prediction behavior can vary across local temporal conditions and that historically relevant contexts can provide useful information for subsequent prediction adjustment (Chen et al., 2024). From this perspective, historical residuals can be viewed not merely as error observations used to characterize prediction quality, but also as records of how a fixed forecasting model previously deviated under particular input conditions.

A.3 Extended Background on Test-Time Adaptation and Prediction Calibration

Test-time adaptation (TTA) aims to improve the deployment-time behavior of a pretrained model using information available during inference (Wang et al., 2021; 2022; Niu et al., 2022; 2023; Zhang et al., 2022a). Across other machine-learning settings, representative strategies include adapting normalization statistics, updating selected model parameters, minimizing prediction entropy, or optimizing auxiliary objectives on test inputs. These approaches allow a pretrained source model to respond to changes encountered after offline training without conventional retraining from scratch.

Applying this paradigm to time-series forecasting introduces an important information constraint: the future target associated with the current forecast is not available when the prediction is issued. Forecasting adaptation methods therefore differ in what deployment-time information they assume to be available. Depending on the setting, they may exploit newly observed values, delayed feedback, partially observed future targets, self-supervised signals, or historically relevant contexts.

Recent forecasting methods instantiate these ideas in different ways. TAFAS addresses non-stationary forecasting through periodicity-aware test-time adaptation and a gated calibration mechanism, using partially observed test-time ground truth as observations become available (Kim et al., 2025). SOLID instead performs sample-level contextualized adaptation: historically similar samples are selected for the current test context and used to construct a local adaptation set for adapting the prediction layer (Chen et al., 2024).

These methods illustrate how deployment-time predictions can be adjusted after offline model training by exploiting additional deployment-time or historical information. However, they differ in the information boundary, the model components being adapted, and whether gradient-based optimization is required during deployment. TTC considers a prediction-level calibration setting in which the forecasting backbone remains fixed and residual information is obtained only from historical windows whose outcomes have already been observed. The unknown future labels of the current forecasting window are not used during calibration.

A.4 Similarity-Based and Retrieval-Inspired Time-Series Methods

Similarity-aware representations and retrieval-inspired techniques have been explored in time-series representation learning, forecasting, and contextualized adaptation (Yue et al., 2022; Woo et al., 2022; Tonekaboni et al., 2021; Li et al., 2023; Dong et al., 2023; Cheng et al., 2026). A common motivation is that locally related temporal patterns can provide useful information beyond that captured by a single global forecasting rule.

Representation-learning approaches construct embeddings that aim to preserve relationships among time-series segments with related shapes, trends, periodic structures, or local dynamics. These rep-

resentations can subsequently support similarity search or retrieval of historical windows that are locally relevant to a current query. Context-aware forecasting methods likewise use historical similarity to identify subsets of previous observations that may be more relevant to the current forecasting context (Chen et al., 2024).

Defining useful similarity for time-series windows is itself non-trivial. Absolute level, scale, normalized temporal shape, local trend, periodic phase, and short-term variation may all affect whether two historical windows should be considered related. Moreover, similarity in the observed input does not guarantee identical future behavior or identical forecasting error. Two windows with similar observed patterns may still produce residual trajectories with different magnitudes or directions.

These considerations motivate careful retrieval and aggregation of historical evidence. In the setting considered in this work, historical inputs are used to retrieve residual trajectories generated by the same frozen forecaster on previously observed windows. The retrieved residuals are then combined to estimate a prediction-level correction rather than being treated as individually transferable errors. The residual-strategy ablation in Section 4.3 empirically compares several ways of using such historical information, including global residual reuse, random retrieval, single-neighbor retrieval, multi-neighbor aggregation, and the complete TTC procedure.

The TTC estimator is also related to classical local non-parametric estimation in the sense that observations near a query in a chosen representation space are used to estimate a context-dependent quantity. The distinction lies in the estimation target and deployment setting: TTC estimates the residual trajectory of an already trained and frozen forecaster, rather than directly estimating the future target or updating forecasting-model parameters. This connection motivates the conditional-residual formulation detailed in Appendix B.1.

B Additional Method and Deployment Details

This section provides additional details on the statistical interpretation, deployment protocol, memory construction, and computational properties of TTC. We first formalize TTC as conditional residual estimation, then clarify the deployment protocol and information boundary, and finally discuss memory subsampling, computational complexity, and practical retrieval efficiency.

B.1 Conditional-Residual Interpretation

We provide a population-level interpretation of TTC as conditional residual estimation. Consider a frozen forecasting backbone and a generic forecasting instance. For variable c, let

$$\mathbf{R}_c = \mathbf{Y}_{:,c} - \hat{\mathbf{Y}}_{:,c} \in \mathbb{R}^H \tag{15}$$

denote the H-step true-minus-predicted residual trajectory, and let $\mathbf{Z}_c$ denote an observable representation of the corresponding look-back context. The conditional residual function is

$$\mathbf{b}_c^*(\mathbf{z}) = \mathbb{E}\left[\mathbf{R}_c \mid \mathbf{Z}_c = \mathbf{z}\right]. \tag{16}$$

The quantity $\mathbf{b}_c^*(\mathbf{z})$ captures the component of the frozen forecaster's residual that is predictable, in the conditional-mean sense, from the observed temporal context. This interpretation does not assume that the entire forecasting error is predictable. Instead, the residual admits the decomposition

$$\mathbf{R}_c = \mathbf{b}_c^*(\mathbf{Z}_c) + \boldsymbol{\varepsilon}_c, \qquad \mathbb{E}\left[\boldsymbol{\varepsilon}_c \mid \mathbf{Z}_c\right] = \mathbf{0}, \tag{17}$$

where $\boldsymbol{\varepsilon}_c$ is the residual component not explained by the chosen context representation.

Under squared error, this decomposition gives a direct population-level interpretation of residual calibration. Define the Base risk for variable c as

$$\mathcal{R}_{\mathrm{Base},c} = \mathbb{E}\left[\|\mathbf{R}_c\|_2^2\right]. \tag{18}$$

If the exact conditional residual function were available, an oracle corrected prediction would be

$$\hat{\mathbf{Y}}_{:,c}^{\mathrm{oracle}} = \hat{\mathbf{Y}}_{:,c} + \mathbf{b}_c^*(\mathbf{Z}_c), \tag{19}$$

with population risk

$$\mathcal{R}_{\mathrm{oracle},c} = \mathbb{E}\left[\|\mathbf{R}_c - \mathbf{b}_c^*(\mathbf{Z}_c)\|_2^2\right]. \tag{20}$$

By the orthogonality property of conditional expectation,

$$\mathcal{R}_{\mathrm{Base},c} - \mathcal{R}_{\mathrm{oracle},c} = \mathbb{E}\left[\|\mathbf{b}_c^*(\mathbf{Z}_c)\|_2^2\right] \geq 0. \tag{21}$$

Equation (21) characterizes the population-level opportunity for residual calibration. If $\mathbf{b}_c^*(\mathbf{Z}_c) = \mathbf{0}$ almost surely, then the chosen observable context contains no conditional-mean residual signal beyond the Base forecast. Conversely, a nonzero conditional residual indicates that part of the forecasting error varies systematically with the observed context and can, in principle, be exploited by prediction-level calibration.

TTC approximates this conditional residual from a finite historical memory. For the current query $\mathbf{q}_{t,c}$, TTC uses

$$\hat{\mathbf{b}}_{t,c} = \sum_{i \in \mathcal{N}_K^{(c)}(t)} w_{i,c} \mathbf{R}_{i,:,c}, \tag{22}$$

which can be viewed as a local non-parametric approximation to $\mathbf{b}_c^*(\mathbf{Z}_{t,c})$. Multiple neighbors reduce dependence on any single historical residual trajectory, while variable-wise retrieval allows different variables to estimate their conditional residuals from different local historical neighborhoods.

Interpretation of residual shrinkage. The same view also provides an interpretation of the correction strength λ. Suppose, for analysis, that the exact conditional residual $\mathbf{b}_c^*(\mathbf{Z}_c)$ were available and consider the oracle shrinkage correction

$$\hat{\mathbf{Y}}_{:,c}^{(\lambda)} = \hat{\mathbf{Y}}_{:,c} + \lambda \mathbf{b}_c^*(\mathbf{Z}_c). \tag{23}$$

Its population squared-error risk is

$$\mathcal{R}_c(\lambda) = \mathbb{E}\left[\|\mathbf{R}_c - \lambda \mathbf{b}_c^*(\mathbf{Z}_c)\|_2^2\right]. \tag{24}$$

Using the conditional-expectation identity

$$\mathbb{E}\left[\mathbf{R}_c^\top \mathbf{b}_c^*(\mathbf{Z}_c)\right] = \mathbb{E}\left[\|\mathbf{b}_c^*(\mathbf{Z}_c)\|_2^2\right], \tag{25}$$

the reduction relative to the Base risk is

$$\mathcal{R}_{\mathrm{Base},c} - \mathcal{R}_c(\lambda) = (2\lambda - \lambda^2)\mathbb{E}\left[\|\mathbf{b}_c^*(\mathbf{Z}_c)\|_2^2\right]. \tag{26}$$

Thus, in the oracle population setting, $0 < \lambda < 2$ yields a non-negative reduction in squared-error risk whenever the conditional residual signal is nonzero, and $\lambda = 1$ maximizes this oracle reduction. This is not a finite-sample performance guarantee for TTC: in practice, the unknown conditional residual is replaced by the finite-memory estimate $\hat{\mathbf{b}}_{t,c}$, which can contain retrieval mismatch and estimation noise. The shrinkage factor λ therefore limits the influence of the estimated residual, while the correction-ratio fallback provides an additional safeguard against excessively large finite-memory corrections.

B.2 Retrieval Representation Details

We provide the implementation details of the variable-wise retrieval representations used by TTC. Let $\mathbf{x}_c = [x_{1,c}, \ldots, x_{L,c}]^\top \in \mathbb{R}^L$ denote the observed look-back sequence of variable c. Its window-level mean and population standard deviation are

$$\mu_c = \frac{1}{L}\sum_{\ell=1}^{L} x_{\ell,c}, \qquad \sigma_c = \sqrt{\frac{1}{L}\sum_{\ell=1}^{L}(x_{\ell,c} - \mu_c)^2}, \tag{27}$$

and the normalized temporal shape is

$$z_{\ell,c} = \frac{x_{\ell,c} - \mu_c}{\max(\sigma_c, 10^{-6})}, \qquad \ell = 1, \ldots, L. \tag{28}$$

The statistical block is therefore $\mathbf{u}_{\mathrm{stat},c} = [\mu_c, \sigma_c] \in \mathbb{R}^2$, while the shape block is $\mathbf{u}_{\mathrm{shape},c} = [z_{1,c}, \ldots, z_{L,c}] \in \mathbb{R}^L$.

For the first-order-difference block, we first compute

$$r_{\ell,c} = x_{\ell+1,c} - x_{\ell,c}, \qquad \ell = 1, \dots, L-1, \tag{29}$$

and standardize this difference sequence within the same look-back window:

$$d_{\ell,c} = \frac{r_{\ell,c} - \mu_{\Delta,c}}{\max(\sigma_{\Delta,c}, 10^{-6})}, \qquad \ell = 1, \dots, L-1, \tag{30}$$

where $\mu_{\Delta,c}$ and $\sigma_{\Delta,c}$ are the mean and population standard deviation of $\{r_{\ell,c}\}_{\ell=1}^{L-1}$. This gives $\mathbf{u}_{\mathrm{diff},c} = [d_{1,c}, \dots, d_{L-1,c}] \in \mathbb{R}^{L-1}$.

Before concatenation, each active feature block is normalized to unit ℓ_2 norm,

$$\bar{\mathbf{u}}_{m,c} = \frac{\mathbf{u}_{m,c}}{\|\mathbf{u}_{m,c}\|_2 + \epsilon}, \qquad m \in \{\mathrm{stat}, \mathrm{shape}, \mathrm{diff}\}. \tag{31}$$

V1 uses the raw look-back values, V2 combines the statistical and normalized-shape blocks, and V3 additionally includes the normalized first-order-difference block. Accordingly,

$$\phi_c^{\mathrm{V1}}(\mathbf{X}) = \mathbf{x}_c, \tag{32}$$

$$\phi_c^{\mathrm{V2}}(\mathbf{X}) = \mathrm{Concat}\left(\alpha_{\mathrm{stat}}\bar{\mathbf{u}}_{\mathrm{stat},c}, \alpha_{\mathrm{shape}}\bar{\mathbf{u}}_{\mathrm{shape},c}\right), \tag{33}$$

$$\phi_c^{\mathrm{V3}}(\mathbf{X}) = \mathrm{Concat}\left(\alpha_{\mathrm{stat}}\bar{\mathbf{u}}_{\mathrm{stat},c}, \alpha_{\mathrm{shape}}\bar{\mathbf{u}}_{\mathrm{shape},c}, \alpha_{\mathrm{diff}}\bar{\mathbf{u}}_{\mathrm{diff},c}\right). \tag{34}$$

For V3, the feature dimension before the final embedding normalization is therefore $D_c = 2 + L + (L-1) = 2L+1$. The concatenated representation is subsequently normalized as in Eqs. (3) and (7) before squared-Euclidean retrieval. This construction keeps the retrieval representation independent of backbone-specific hidden states while preserving complementary information about local level, normalized temporal shape, and short-term changes.

B.3 Deployment Protocol and Information Boundary

TTC follows a fixed protocol that separates forecasting model training, residual-memory construction, and final test-time calibration. The forecasting backbone is first trained using its original forecasting objective, and a checkpoint $f_{\theta^\star}$ is selected according to validation performance. Once selected, this checkpoint remains frozen throughout residual-memory construction and final test evaluation.

The residual memory bank is constructed only from historical labeled windows whose corresponding future observations have already become available. For each historical pair $(\mathbf{X}_i, \mathbf{Y}_i)$, the frozen forecaster produces

$$\hat{\mathbf{Y}}_i = f_{\theta^\star}(\mathbf{X}_i), \qquad \mathbf{R}_i = \mathbf{Y}_i - \hat{\mathbf{Y}}_i, \tag{35}$$

where the variable-wise retrieval representations derived from $\mathbf{X}_i$ are stored together with the residual trajectory $\mathbf{R}_i$ in the memory bank $\mathcal{M}$. After construction, $\mathcal{M}$ remains fixed during final test evaluation.

For a current test window $\mathbf{X}_t$, TTC uses only the observed look-back sequence, the frozen-backbone prediction, the pre-built residual memory, and calibration hyperparameters fixed before final test evaluation. The key information available to the calibration step can therefore be summarized as

$$\mathcal{I}_t = \left\{\mathbf{X}_t, \hat{\mathbf{Y}}_t, \mathcal{M}, K, \tau, \lambda, \delta_r\right\}, \tag{36}$$

whereas the unknown future target $\mathbf{Y}_t$ is not included in $\mathcal{I}_t$.

Accordingly, the future labels of the current test window are never used for similarity retrieval, residual aggregation, correction-ratio evaluation, or final prediction. TTC also performs no gradient-based parameter update during calibration. The forecasting backbone, residual memory bank, and calibration configuration therefore remain fixed throughout final test evaluation.

B.4 Memory Density and Historical Subsampling

The residual memory bank does not need to retain every available historical window. A fixed memory stride can be used to subsample historical candidates before constructing the final memory. Let N denote the number of available historical windows and $N_s \leq N$ the number retained after subsampling. The resulting memory can be written as

$$\mathcal{M}_s = \{(\mathbf{E}_i, \mathbf{R}_i)\}_{i \in \mathcal{S}}, \qquad |\mathcal{S}| = N_s, \tag{37}$$

where $\mathcal{S}$ denotes the retained historical indices.

A denser memory provides more candidate temporal contexts for retrieval, but increases both storage and similarity-search cost. Conversely, a larger memory stride reduces the number of stored items and therefore decreases retrieval and storage overhead, although an excessively sparse memory may remove useful historical patterns. Memory density therefore provides a direct trade-off between retrieval coverage and storage/computational cost.

Importantly, historical subsampling changes only the set of candidate memory items. It does not modify the TTC calibration formulation: every retained item still contains the same variable-wise retrieval representation and its corresponding true-minus-predicted residual trajectory.

B.5 Storage and Computational Complexity

We next analyze the additional storage and computation introduced by TTC. Let N denote the number of memory items, C the number of variables, D_c the dimensionality of each variable-wise retrieval representation, L the look-back length, H the prediction horizon, and K the number of retrieved neighbors.

Each memory item stores C variable-wise representations of dimension D_c together with a residual tensor of size $H \times C$. The storage complexity of the complete residual memory is therefore

$$O\left(NC(D_c + H)\right). \tag{38}$$

When memory subsampling retains only N_s historical windows, this becomes

$$O\left(N_s C(D_c + H)\right), \qquad N_s \leq N. \tag{39}$$

For each test query, constructing the lightweight variable-wise retrieval representation requires $O(LC)$ operations. TTC then compares each query representation with the corresponding historical representations. Under exhaustive retrieval, computing distances over all stored memory items requires

$$O(NCD_c) \tag{40}$$

operations per test window, or

$$O(N_s C D_c) \tag{41}$$

when a subsampled memory is used. Top-K selection introduces an additional implementation-dependent selection cost; for example, a standard heap-based implementation requires $O(CN_s \log K)$ operations.

After the top-K neighbors are retrieved, their H-step residual trajectories are aggregated independently for each variable, resulting in

$$O(KHC) \tag{42}$$

additional computation. The correction-ratio fallback operates on the resulting candidate correction and Base prediction for all variables, requiring

$$O(HC). \tag{43}$$

Thus, the post-retrieval calibration stage has complexity

$$O(KHC) + O(HC) = O((K+1)HC). \tag{44}$$

When $N_s \gg K$, exhaustive similarity search is typically the dominant TTC-specific computation, while residual aggregation and fallback scale only with the retrieved neighborhood and prediction horizon.

B.6 Practical Retrieval Efficiency

The practical retrieval cost of TTC is mainly determined by the number of stored historical items and the dimensionality of the variable-wise representation. Several implementation choices can reduce this cost without changing the prediction-level calibration formulation.

First, memory stride or other fixed historical subsampling reduces the number of stored items from N to N_s, lowering both memory consumption and exhaustive retrieval cost. Second, distance computations across multiple variables and queries can be batched to exploit parallel numerical operations. For substantially larger residual memories, exhaustive search could also be replaced by approximate nearest-neighbor indexing while retaining the same downstream residual aggregation and correction-ratio mechanism.

TTC does not require backpropagation during inference. The forecasting backbone performs only a forward pass, while the additional calibration operations consist of retrieval-representation construction, historical similarity search, weighted residual aggregation, and correction-ratio evaluation. Under a fixed evaluation protocol, the same pre-built residual memory can be reused across all test queries.

The complexity analysis above describes how TTC scales with memory size, representation dimensionality, the number of variables, and prediction horizon. Actual wall-clock latency and memory footprint additionally depend on the forecasting backbone, dataset dimensionality, memory density, retrieval implementation, batching strategy, and hardware configuration. Accordingly, the analysis above characterizes asymptotic scaling rather than claiming a hardware-independent estimate of wall-clock latency or memory footprint.

B.7 Empirical Efficiency and Scalability

We further evaluate the practical inference overhead of TTC over the complete 160-setting evaluation grid. Base and TTC use the same frozen forecasting checkpoint and test inputs within each setting. End-to-end latency excludes data loading, checkpoint loading, and one-time residual-memory construction. Measurements are performed on NVIDIA A40 GPUs with 20 warm-up iterations followed by 100 timed iterations. The complete benchmark contains 160 settings for batch size one and the same 160 settings for batch size 32, with no failed settings. The scalability benchmark uses the TTC configuration recorded for each evaluated setting; otherwise, the default configuration in Appendix C.2 applies unless explicitly stated.

Figure 4(a) compares the end-to-end Base and TTC latency on Weather with $H = 96$ across five forecasting backbones. TTC introduces additional computation for query-representation construction, similarity retrieval, residual aggregation, and correction-ratio evaluation. The additional latency remains at the millisecond scale across the evaluated backbones. Relative slowdown is naturally more pronounced for lightweight models such as DLinear because their Base forward pass is already extremely fast, whereas heavier forecasting backbones exhibit a smaller relative increase.

Figure 4(b) summarizes scalability across all 160 model–dataset–horizon settings. At batch size one, the median additional latency is 0.876 ms/query and the median slowdown is $1.410\times$. At batch size 32, the median additional latency is 0.862 ms/batch and the median slowdown is $1.240\times$. The median residual-memory footprint over the full grid is 61.5 MB; Solar forms the most demanding regime, with a median memory footprint of 2808.7 MB and a maximum observed footprint of 5632.2 MB. Larger residual memories are strongly associated with higher retrieval overhead, which

is consistent with the exhaustive variable-wise retrieval complexity analyzed in Appendix B.5. Small fluctuations around zero are attributable to runtime measurement variability rather than algorithmic acceleration.

Overall, the empirical results complement the asymptotic analysis: TTC typically adds millisecond-scale prediction-level overhead, while storage and retrieval cost become more important in high-dimensional settings with substantially larger residual memories.

C Additional Experimental Details and Complete Results

This section complements the main experimental section with additional protocol and reproducibility details that are omitted from the main text for compactness. We first clarify the shared-Base evaluation protocol, the default TTC configuration, and the configuration coverage of the two comparison baselines. We then report the complete horizon-wise results for all five forecasting backbones. Unless otherwise stated, the same validation-selected forecasting checkpoint is used as the common starting point for Base, TAFAS, SOLID, and TTC within each model–dataset–horizon setting.

C.1 Evaluation and Shared-Base Protocol

All datasets are divided chronologically into training, validation, and test subsets. Forecasting checkpoints are selected according to validation performance, and the final test split is used only for reporting. For each backbone–dataset–horizon setting, TAFAS, SOLID, and TTC start from the same validation-selected forecasting checkpoint and are evaluated on the same test windows using the same forecasting metrics. This shared-Base protocol controls differences in source-model quality and allows the comparison to focus on the additional effect introduced by each test-time adaptation or calibration mechanism.

The shared-Base protocol aligns the source forecasting checkpoint and evaluation data, but does not imply identical deployment-time information assumptions across methods. TAFAS, SOLID, and TTC differ in the information available during adaptation or calibration and in whether model parameters are updated; these differences follow the respective method formulations described in Section 2.

For the horizon-averaged rows, MSE and MAE are first averaged over the four prediction horizons, i.e., $H \in \{96, 192, 336, 720\}$ for all datasets except Illness, which uses $H \in \{24, 36, 48, 60\}$. The relative improvement is then computed from the corresponding averaged errors,

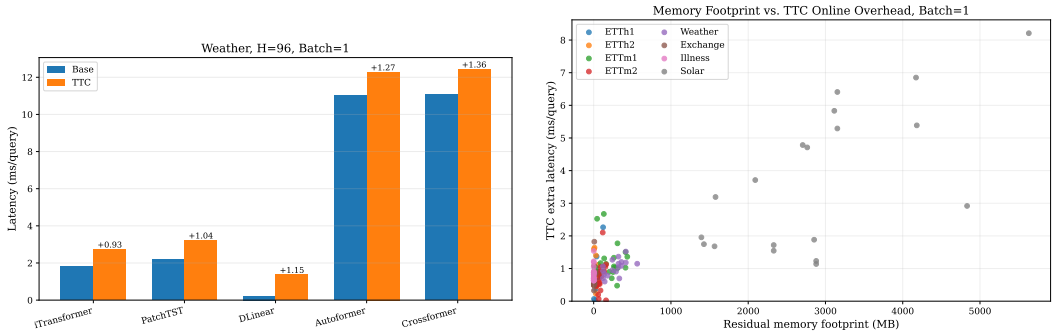
$$\text{Imp.} = \frac{\bar{m}_{\text{Base}} - \bar{m}_{\text{Method}}}{\bar{m}_{\text{Base}}} \times 100\%, \tag{45}$$

where $\bar{m}$ denotes the horizon-averaged MSE or MAE. Thus, the reported Avg Imp. values are computed after horizon averaging rather than by averaging four horizon-wise improvement percentages. Positive values indicate lower error than the shared Base, whereas negative values indicate degradation.

C.2 Default TTC Configuration

Unless otherwise stated, TTC uses the following default configuration. The default retrieval representation is V3, which combines statistical properties, normalized temporal shape, and first-order differences with feature weights $(\alpha_{\text{stat}}, \alpha_{\text{shape}}, \alpha_{\text{diff}}) = (0.5, 1.0, 0.7)$.

After checkpoint selection, the default residual memory is constructed from historical windows in the training and validation splits with memory stride $s = 8$. For each query variable, TTC retrieves $K = 20$ historical neighbors, aggregates their residual trajectories using temperature $\tau = 0.10$, scales the resulting compensation by correction strength $\lambda = 0.20$, and applies the correction-ratio fallback with threshold $\delta_r = 0.40$. When an experiment explicitly specifies a different setting, the reported setting is used; otherwise, the default values above apply.



(a) Online latency on Weather with $H = 96$.

(b) Scalability across the complete evaluation grid.

Figure 4: Empirical inference efficiency and scalability of TTC. (a) End-to-end online latency of the frozen Base forecaster and TTC with batch size one on Weather with $H = 96$ across five forecasting backbones. Numbers above the TTC bars denote the additional latency relative to the corresponding Base. (b) Relationship between residual-memory footprint and additional TTC latency across the complete 160 model–dataset–horizon settings; each point denotes one evaluation setting. Data loading, checkpoint loading, and one-time residual-memory construction are excluded from online latency.

C.3 Baseline Implementation

We implement TAFAS and SOLID following their published methods and official codebases, and evaluate them within the same forecasting pipeline used for TTC. For each backbone–dataset–horizon setting, all methods start from the same validation-selected forecasting checkpoint and are evaluated on the same test data.

Where released hyperparameter settings are directly applicable, we use the corresponding official configurations. For settings not explicitly covered by the released configurations, we apply the same adaptation procedure with fixed predefined settings rather than tuning configurations separately for individual test cases. No baseline configuration is selected according to final test performance.

This protocol is intended to compare the additional effect of different test-time adaptation or calibration mechanisms on the same underlying forecasting model, rather than to reproduce the exact experimental setup of each original baseline paper.

C.4 Complete Horizon-Wise Comparison

For direct comparison, the complete results are organized by forecasting backbone. Each dataset is shown as a six-row block: four prediction horizons, an Avg. row, and an Avg Imp. row. The Avg. row reports MSE and MAE averaged over the four horizons, while the Avg Imp. row reports the corresponding relative improvements defined in Eq. (45). MSE and MAE are reported to three decimal places, and average improvements are reported to two decimal places. Boldface indicates the best result according to the unrounded values. Lower MSE and MAE and higher improvements are better.

Table 3: Complete horizon-wise comparison for iTransformer under the shared-Base protocol.

Dataset	H	Base		TAFAS		SOLID		TTC (Ours)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.386	0.405	0.385	0.405	0.380	0.401	**0.377**	**0.401**
	192	0.441	0.436	0.435	0.437	0.435	0.433	**0.429**	**0.433**
	336	0.491	0.461	0.483	0.462	0.480	**0.456**	**0.471**	0.457
	720	0.495	0.487	0.501	0.495	0.488	0.484	**0.485**	**0.483**
	Avg.	0.453	0.447	0.451	0.450	0.446	0.444	**0.441**	**0.443**
	Avg Imp.	–	–	+0.52%	−0.61%	+1.67%	+0.79%	+2.80%	+0.84%
ETTh2	96	0.300	0.350	0.302	0.351	0.300	0.350	**0.293**	**0.344**
	192	0.380	0.398	0.382	0.399	0.380	0.398	**0.370**	**0.391**
	336	0.423	0.432	0.420	0.431	0.423	0.432	**0.410**	**0.423**
	720	0.431	0.447	0.436	0.451	0.431	0.447	**0.409**	**0.432**
	Avg.	0.384	0.407	0.385	0.408	0.384	0.407	**0.370**	**0.397**
	Avg Imp.	–	–	−0.34%	−0.29%	+0.01%	+0.01%	+3.43%	+2.31%
ETTm1	96	0.343	0.377	0.333	0.373	0.342	0.377	**0.316**	**0.361**
	192	0.381	0.395	0.368	0.392	0.380	0.395	**0.360**	**0.386**
	336	0.424	0.421	0.406	0.421	0.423	0.421	**0.399**	**0.413**
	720	0.486	0.456	0.466	0.455	0.486	0.456	**0.461**	**0.449**
	Avg.	0.408	0.412	0.393	0.410	0.408	0.412	**0.384**	**0.402**
	Avg Imp.	–	–	+3.67%	+0.42%	+0.04%	+0.02%	+5.97%	+2.44%
ETTm2	96	0.182	0.266	0.182	0.267	0.182	0.266	**0.176**	**0.260**
	192	0.254	0.313	0.253	0.315	0.254	0.313	**0.247**	**0.310**
	336	0.316	0.352	0.314	0.354	0.316	0.352	**0.308**	**0.350**
	720	0.412	0.407	0.425	0.417	0.412	**0.407**	**0.404**	0.419
	Avg.	0.291	0.335	0.294	0.338	0.291	**0.335**	**0.284**	0.335
	Avg Imp.	–	–	−0.91%	−1.06%	0.00%	0.00%	+2.47%	−0.07%
Weather	96	0.175	0.214	0.166	0.213	0.175	0.214	**0.163**	**0.209**
	192	0.224	0.257	0.210	0.254	0.224	0.257	**0.207**	**0.254**
	336	0.281	0.298	0.263	0.296	0.281	0.298	**0.257**	**0.293**
	720	0.359	0.350	0.342	0.349	0.359	0.350	**0.325**	**0.344**
	Avg.	0.260	0.280	0.245	0.278	0.260	0.280	**0.238**	**0.275**
	Avg Imp.	–	–	+5.50%	+0.70%	0.00%	0.00%	+8.53%	+1.79%
Exchange	96	0.087	0.207	**0.084**	**0.206**	0.087	0.207	0.085	0.207
	192	0.184	0.308	0.175	**0.302**	0.184	0.308	**0.168**	0.303
	336	0.330	0.417	0.297	0.399	0.330	0.417	**0.268**	**0.398**
	720	0.858	0.700	0.835	0.690	0.858	0.700	**0.536**	**0.589**
	Avg.	0.365	0.408	0.348	0.399	0.365	0.408	**0.264**	**0.374**
	Avg Imp.	–	–	+4.70%	+2.22%	0.00%	0.00%	+27.57%	+8.32%
Illness	24	2.368	1.035	2.367	1.034	2.499	1.033	**1.893**	**0.849**
	36	2.280	1.022	2.277	1.021	2.305	1.013	**1.897**	**0.876**
	48	2.323	1.034	2.320	1.033	2.312	1.023	**1.913**	**0.909**
	60	2.379	1.056	2.377	1.056	2.362	1.053	**1.926**	**0.919**
	Avg.	2.337	1.037	2.335	1.036	2.369	1.030	**1.907**	**0.888**
	Avg Imp.	–	–	+0.10%	+0.10%	−1.37%	+0.63%	+18.41%	+14.32%
Solar	96	0.202	0.237	0.201	0.241	0.201	0.236	**0.190**	**0.229**
	192	0.237	0.263	0.237	0.267	0.237	0.262	**0.223**	**0.254**
	336	0.249	0.273	0.248	0.278	0.248	0.273	**0.227**	**0.266**
	720	0.251	0.276	0.251	0.280	0.251	0.276	**0.234**	**0.270**
	Avg.	0.235	0.262	0.234	0.267	0.234	0.262	**0.218**	**0.255**
	Avg Imp.	–	–	+0.24%	−1.62%	+0.15%	+0.15%	+6.95%	+2.89%

Table 3 shows a broadly consistent benefit of TTC for the variable-centric iTransformer backbone. TTC achieves the lowest horizon-averaged MSE on all eight datasets and the lowest averaged MAE on seven of them, indicating that the residual-memory mechanism remains effective across datasets with different temporal characteristics and forecasting difficulty. The largest gains appear on Exchange and Illness, whereas the ETT benchmarks generally show more moderate improvements. This contrast suggests that TTC does not impose a uniform correction across datasets; instead, its benefit depends on how much reusable, context-dependent residual structure remains after the frozen backbone has produced its forecast. ETTm2 is the main exception in MAE, where TTC is essentially unchanged from the Base and SOLID is slightly better according to the unrounded value. Overall, the iTransformer results support historical residual retrieval as a complementary calibration signal while also showing that the achievable gain remains data dependent.

Table 4: Complete horizon-wise comparison for PatchTST under the shared-Base protocol.

Dataset	H	Base		TAFAS		SOLID		TTC (Ours)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.377	0.399	0.379	0.401	**0.371**	**0.393**	0.374	0.398
	192	0.414	0.421	0.415	0.423	**0.408**	**0.417**	0.410	0.420
	336	0.444	0.443	0.445	0.445	0.440	**0.440**	**0.438**	0.442
	720	0.458	0.472	0.458	0.475	0.453	0.469	**0.452**	**0.468**
	Avg.	0.423	0.434	0.424	0.436	0.418	**0.430**	**0.418**	0.432
	Avg Imp.	–	–	−0.27%	−0.45%	+1.16%	+0.95%	+1.17%	+0.39%
ETTh2	96	0.276	0.338	0.277	0.339	0.277	0.338	**0.271**	**0.337**
	192	0.342	0.378	0.342	0.379	0.342	0.378	**0.334**	**0.375**
	336	0.371	0.402	0.371	0.404	0.371	0.402	**0.365**	**0.399**
	720	0.391	0.428	0.389	0.427	0.391	0.428	**0.376**	**0.421**
	Avg.	0.345	0.387	0.345	0.387	0.345	0.387	**0.337**	**0.383**
	Avg Imp.	–	–	+0.01%	−0.10%	0.00%	0.00%	+2.48%	+0.97%
ETTm1	96	0.291	0.343	0.291	0.345	0.290	0.341	**0.284**	**0.337**
	192	0.337	0.372	0.337	0.373	0.337	0.372	**0.330**	**0.367**
	336	0.366	0.392	0.365	0.391	0.367	0.392	**0.359**	**0.387**
	720	0.420	0.425	0.420	0.425	0.420	0.424	**0.413**	**0.421**
	Avg.	0.353	0.383	0.353	0.384	0.353	0.382	**0.346**	**0.378**
	Avg Imp.	–	–	+0.07%	−0.15%	−0.01%	+0.20%	+1.96%	+1.29%
ETTm2	96	0.164	0.253	0.164	0.255	0.164	0.253	**0.160**	**0.250**
	192	0.223	0.295	0.222	0.295	0.223	0.295	**0.217**	**0.290**
	336	0.277	0.329	0.276	0.330	0.277	0.329	**0.271**	**0.326**
	720	0.366	0.383	0.362	0.383	0.366	0.384	**0.358**	**0.380**
	Avg.	0.258	0.315	0.256	0.316	0.258	0.315	**0.252**	**0.311**
	Avg Imp.	–	–	+0.47%	−0.25%	−0.04%	−0.01%	+2.33%	+1.17%
Weather	96	0.150	0.197	0.150	0.200	0.150	0.197	**0.145**	**0.195**
	192	0.195	0.240	0.193	0.244	0.195	0.240	**0.189**	**0.238**
	336	0.247	0.282	0.245	0.285	0.247	0.282	**0.239**	**0.277**
	720	0.325	0.337	0.325	0.354	0.325	0.337	**0.311**	**0.330**
	Avg.	0.229	0.264	0.228	0.271	0.229	0.264	**0.221**	**0.260**
	Avg Imp.	–	–	+0.47%	−2.51%	+0.02%	+0.03%	+3.69%	+1.48%
Exchange	96	0.096	0.219	0.094	0.217	0.096	0.219	**0.091**	**0.216**
	192	0.200	0.322	0.185	0.313	0.201	0.322	**0.181**	**0.312**
	336	0.355	0.436	0.313	0.424	0.356	0.437	**0.296**	**0.422**
	720	0.889	0.701	0.885	0.699	0.889	0.701	**0.642**	**0.603**
	Avg.	0.385	0.420	0.369	0.413	0.385	0.420	**0.303**	**0.388**
	Avg Imp.	–	–	+4.10%	+1.49%	−0.04%	−0.01%	+21.42%	+7.53%
Illness	24	2.056	0.908	2.054	0.907	2.314	0.912	**1.903**	**0.850**
	36	1.970	0.938	1.966	0.935	1.998	0.918	**1.684**	**0.829**
	48	1.704	0.882	1.701	0.880	1.703	0.868	**1.552**	**0.812**
	60	2.016	0.964	2.015	0.963	2.005	0.959	**1.740**	**0.870**
	Avg.	1.937	0.923	1.934	0.921	2.005	0.914	**1.720**	**0.840**
	Avg Imp.	–	–	+0.13%	+0.18%	−3.53%	+0.96%	+11.20%	+9.00%
Solar	96	0.213	0.257	0.220	0.268	0.211	0.255	**0.202**	**0.247**
	192	0.247	0.278	0.253	0.286	0.246	0.277	**0.231**	**0.272**
	336	0.261	0.288	0.264	0.292	0.261	0.287	**0.241**	**0.279**
	720	0.260	0.286	0.262	0.295	0.260	0.286	**0.243**	**0.279**
	Avg.	0.246	0.277	0.250	0.285	0.245	0.276	**0.229**	**0.269**
	Avg Imp.	–	–	−1.74%	−2.85%	+0.39%	+0.31%	+6.70%	+2.87%

Table 4 presents a similarly stable pattern for the patch-based PatchTST backbone. TTC obtains the best horizon-averaged MSE on all eight datasets and the best averaged MAE on seven datasets, with ETTh1 favoring SOLID in MAE. The improvements are especially clear on Exchange and Illness, while several ETT settings exhibit smaller but still broadly consistent gains. Because PatchTST already captures local temporal structure through patch-based representations, the additional improvement suggests that external residual memory provides information that is complementary to the representation learned by the frozen forecaster. Importantly, TTC obtains this benefit without changing PatchTST parameters or introducing a backbone-specific adaptation objective. The ETTh1 exception further shows that different test-time mechanisms can remain preferable in individual settings even when TTC provides the strongest overall pattern for this backbone.

Table 5: Complete horizon-wise comparison for DLinear under the shared-Base protocol.

Dataset	H	Base		TAFAS		SOLID		TTC (Ours)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.371	0.394	0.371	0.395	**0.365**	**0.390**	0.369	0.394
	192	0.407	0.418	0.406	0.416	**0.401**	**0.414**	0.404	0.417
	336	0.434	0.437	**0.426**	**0.430**	0.430	0.434	0.429	0.438
	720	0.475	0.493	0.475	**0.484**	0.472	0.491	**0.469**	0.488
	Avg.	0.422	0.436	0.420	**0.431**	**0.417**	0.432	0.418	0.434
	Avg Imp.	–	–	+0.47%	+1.00%	+1.05%	+0.80%	+0.88%	+0.33%
ETTh2	96	0.286	0.348	0.288	0.347	0.286	0.349	**0.282**	**0.344**
	192	0.386	0.420	0.354	0.394	0.371	0.409	**0.341**	**0.386**
	336	0.449	0.458	0.400	0.432	0.432	0.448	**0.363**	**0.407**
	720	0.579	0.541	0.497	0.503	0.557	0.531	**0.493**	**0.492**
	Avg.	0.425	0.442	0.385	0.419	0.412	0.434	**0.370**	**0.407**
	Avg Imp.	–	–	+9.50%	+5.18%	+3.16%	+1.78%	+13.04%	+7.92%
ETTm1	96	0.301	0.346	0.298	0.344	0.300	0.345	**0.282**	**0.338**
	192	0.338	0.371	0.333	0.365	0.338	0.371	**0.321**	**0.362**
	336	0.370	0.387	0.366	0.386	0.370	0.387	**0.356**	**0.383**
	720	0.425	0.421	0.418	**0.416**	0.425	0.421	**0.405**	0.417
	Avg.	0.358	0.381	0.354	0.378	0.358	0.381	**0.341**	**0.375**
	Avg Imp.	–	–	+1.25%	+0.92%	+0.03%	+0.05%	+4.90%	+1.66%
ETTm2	96	0.166	0.259	0.166	0.257	0.167	0.259	**0.161**	**0.252**
	192	0.232	0.312	**0.223**	**0.298**	0.231	0.312	0.223	0.300
	336	0.298	0.361	0.281	0.339	0.297	0.360	**0.275**	**0.338**
	720	0.411	0.432	0.384	0.404	0.410	0.432	**0.366**	**0.399**
	Avg.	0.277	0.341	0.264	0.324	0.276	0.341	**0.256**	**0.322**
	Avg Imp.	–	–	+4.73%	+4.98%	+0.16%	+0.15%	+7.36%	+5.54%
Weather	96	0.178	0.243	0.171	0.229	0.178	0.243	**0.156**	**0.214**
	192	0.219	0.280	0.212	0.266	0.219	0.280	**0.198**	**0.254**
	336	0.265	0.320	0.257	0.302	0.265	0.319	**0.245**	**0.293**
	720	0.329	0.372	0.320	0.357	0.329	0.372	**0.312**	**0.346**
	Avg.	0.248	0.304	0.240	0.288	0.248	0.303	**0.228**	**0.277**
	Avg Imp.	–	–	+3.24%	+5.04%	+0.06%	+0.10%	+8.06%	+8.91%
Exchange	96	0.081	0.205	**0.078**	0.203	0.081	0.203	0.079	**0.203**
	192	0.158	0.293	**0.151**	0.289	0.158	0.292	0.152	**0.289**
	336	0.339	0.446	0.303	0.430	0.332	0.441	**0.289**	**0.409**
	720	0.752	0.648	0.709	0.631	0.733	0.640	**0.669**	**0.618**
	Avg.	0.332	0.398	0.310	0.388	0.326	0.394	**0.297**	**0.380**
	Avg Imp.	–	–	+6.64%	+2.46%	+1.95%	+0.90%	+10.50%	+4.59%
Illness	24	2.386	1.106	2.370	1.101	2.378	1.059	**1.972**	**0.964**
	36	2.230	1.048	2.218	1.045	2.203	1.024	**1.904**	**0.952**
	48	2.231	1.058	2.224	1.056	2.184	1.039	**1.883**	**0.960**
	60	2.172	1.036	2.168	1.035	2.175	1.036	**1.915**	**0.977**
	Avg.	2.255	1.062	2.245	1.059	2.235	1.040	**1.918**	**0.963**
	Avg Imp.	–	–	+0.43%	+0.29%	+0.89%	+2.12%	+14.92%	+9.33%
Solar	96	0.285	0.372	0.280	0.368	0.284	0.371	**0.234**	**0.310**
	192	0.316	0.393	0.311	0.392	0.315	0.393	**0.259**	**0.332**
	336	0.350	0.413	0.345	0.413	0.349	0.413	**0.278**	**0.348**
	720	0.354	0.410	0.351	0.412	0.354	0.410	**0.285**	**0.357**
	Avg.	0.326	0.397	0.322	0.396	0.326	0.397	**0.264**	**0.337**
	Avg Imp.	–	–	+1.33%	+0.18%	+0.14%	+0.10%	+18.97%	+15.20%

Table 5 shows that the effectiveness of TTC is not limited to deep Transformer-style forecasting models. On the lightweight DLinear backbone, TTC achieves the best horizon-averaged MSE and MAE on seven of the eight datasets; ETTh1 is the main exception, where SOLID gives the best averaged MSE and TAFAS gives the best averaged MAE. The gains on ETTh2, Weather, Illness, and Solar are particularly informative because DLinear uses a deliberately simple linear forecasting mechanism rather than a rich attention-based representation. TTC nevertheless remains effective because its retrieval representation is constructed directly from the observed time series instead of backbone-specific hidden states. This behavior is consistent with the model-agnostic design of TTC: residual memory operates outside the predictor and reuses historical error patterns at the prediction level. The variation across datasets further indicates that the gain depends on whether the frozen predictor leaves systematic residual behavior that can be matched to relevant historical contexts.

Table 6: Complete horizon-wise comparison for Autoformer under the shared-Base protocol.

Dataset	H	Base		TAFAS		SOLID		TTC (Ours)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.470	0.466	0.460	0.462	1.460	0.879	**0.448**	**0.452**
	192	0.473	0.469	0.466	0.470	1.753	0.962	**0.453**	**0.457**
	336	0.511	0.493	0.509	0.495	0.986	0.738	**0.498**	**0.487**
	720	0.500	0.504	0.516	0.511	1.499	0.959	**0.487**	**0.496**
	Avg.	0.488	0.483	0.487	0.484	1.425	0.884	**0.471**	**0.473**
	Avg Imp.	–	–	+0.17%	−0.33%	−191.68%	−83.13%	+3.47%	+2.11%
ETTh2	96	0.367	0.405	0.374	0.410	0.374	0.411	**0.335**	**0.389**
	192	0.447	0.447	0.447	0.448	0.455	0.452	**0.419**	**0.431**
	336	0.478	0.482	0.482	0.484	0.482	0.484	**0.463**	**0.472**
	720	0.453	0.473	0.462	0.476	0.458	0.475	**0.435**	**0.463**
	Avg.	0.436	0.452	0.442	0.455	0.443	0.455	**0.413**	**0.439**
	Avg Imp.	–	–	−1.22%	−0.62%	−1.42%	−0.77%	+5.33%	+2.89%
ETTm1	96	0.476	0.461	0.459	0.453	0.475	0.461	**0.426**	**0.435**
	192	0.565	0.508	0.535	0.494	0.564	0.508	**0.457**	**0.455**
	336	0.546	0.504	0.520	0.491	0.546	0.503	**0.499**	**0.480**
	720	0.544	0.514	0.519	0.499	0.543	0.513	**0.505**	**0.491**
	Avg.	0.533	0.496	0.508	0.484	0.532	0.496	**0.471**	**0.465**
	Avg Imp.	–	–	+4.56%	+2.51%	+0.10%	+0.07%	+11.49%	+6.28%
ETTm2	96	0.232	0.312	0.234	0.314	0.231	0.312	**0.212**	**0.299**
	192	0.332	0.368	**0.308**	**0.359**	0.332	0.368	0.315	0.359
	336	0.359	0.385	0.355	0.386	0.359	0.385	**0.347**	**0.379**
	720	0.425	0.422	0.424	0.424	0.425	**0.422**	**0.413**	0.426
	Avg.	0.337	0.372	0.330	0.371	0.337	0.372	**0.322**	**0.366**
	Avg Imp.	–	–	+1.99%	+0.34%	+0.05%	+0.05%	+4.48%	+1.65%
Weather	96	0.269	0.343	**0.239**	**0.317**	0.269	0.343	0.251	0.327
	192	0.290	0.346	0.268	0.331	0.290	0.346	**0.255**	**0.322**
	336	0.391	0.419	**0.340**	**0.386**	0.391	0.419	0.351	0.393
	720	0.427	0.436	0.391	0.409	0.426	0.436	**0.370**	**0.395**
	Avg.	0.344	0.386	0.310	0.361	0.344	0.386	**0.306**	**0.359**
	Avg Imp.	–	–	+10.01%	+6.58%	+0.02%	+0.01%	+10.92%	+7.02%
Exchange	96	0.141	0.271	0.137	0.269	0.141	0.271	**0.107**	**0.238**
	192	0.523	0.520	0.501	0.509	0.522	0.519	**0.333**	**0.427**
	336	0.691	0.631	0.676	0.621	0.691	0.631	**0.404**	**0.492**
	720	1.170	0.852	1.155	0.844	1.170	0.852	**0.592**	**0.616**
	Avg.	0.631	0.568	0.617	0.561	0.631	0.568	**0.359**	**0.443**
	Avg Imp.	–	–	+2.20%	+1.35%	+0.01%	+0.02%	+43.13%	+21.99%
Illness	24	3.423	1.297	3.415	1.295	3.368	1.282	**2.887**	**1.158**
	36	3.401	1.259	3.399	1.258	3.400	1.261	**2.798**	**1.120**
	48	3.495	1.297	3.492	1.296	3.508	1.292	**2.617**	**1.081**
	60	2.824	1.144	2.823	1.144	2.825	1.149	**2.586**	**1.069**
	Avg.	3.286	1.249	3.282	1.248	3.275	1.246	**2.722**	**1.107**
	Avg Imp.	–	–	+0.11%	+0.09%	+0.32%	+0.26%	+17.16%	+11.41%
Solar	96	0.564	0.510	0.438	0.460	0.572	0.512	**0.370**	**0.424**
	192	0.700	0.632	0.554	0.563	0.708	0.635	**0.412**	**0.483**
	336	0.724	0.642	0.551	0.561	0.723	0.640	**0.354**	**0.408**
	720	0.783	0.684	0.607	0.611	0.787	0.684	**0.371**	**0.434**
	Avg.	0.693	0.617	0.537	0.549	0.697	0.618	**0.377**	**0.438**
	Avg Imp.	–	–	+22.45%	+11.03%	−0.62%	−0.16%	+45.61%	+29.06%

Table 6 shows particularly strong overall gains for the decomposition-based Autoformer backbone. TTC achieves the lowest horizon-averaged MSE and MAE on all eight datasets, with especially pronounced improvements on Solar and Exchange while positive gains remain across the other benchmarks. This pattern is notable because Autoformer relies on decomposition and auto-correlation mechanisms that differ substantially from the variable-centric, patch-based, and linear architectures considered above, providing additional evidence that TTC does not depend on a specific forecasting representation. TAFAS remains competitive on several individual horizons, whereas some extended SOLID configurations, most visibly on ETTh1, exhibit substantial degradation. Because these cases arise under transferred or manually extended configurations rather than directly released per-setting configurations, we retain them for transparency and interpret them as configuration sensitivity under the shared-Base protocol rather than as a general limitation of SOLID. Overall, Autoformer provides a clear example of the value of external residual calibration when a frozen forecaster leaves reusable historical error structure.

Table 7: Complete horizon-wise comparison for Crossformer under the shared-Base protocol.

Dataset	H	Base		TAFAS		SOLID		TTC (Ours)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.473	0.480	**0.382**	**0.414**	0.828	0.715	0.433	0.453
	192	0.417	0.448	**0.406**	**0.438**	0.422	0.454	0.407	0.443
	336	0.437	0.463	**0.422**	**0.448**	0.431	0.460	0.426	0.457
	720	0.531	0.527	0.497	**0.501**	0.501	0.526	**0.494**	0.506
	Avg.	0.465	0.480	**0.427**	**0.450**	0.545	0.539	0.440	0.465
	Avg Imp.	–	–	+8.11%	+6.16%	−17.45%	−12.27%	+5.23%	+3.12%
ETTh2	96	0.604	0.533	0.477	0.477	0.542	0.532	**0.298**	**0.367**
	192	0.798	0.648	0.713	0.614	0.720	0.612	**0.402**	**0.450**
	336	1.289	0.844	0.931	0.713	1.024	0.716	**0.582**	**0.584**
	720	2.110	1.124	1.817	1.031	**1.532**	**0.914**	1.695	1.023
	Avg.	1.200	0.787	0.984	0.709	0.954	0.694	**0.744**	**0.606**
	Avg Imp.	–	–	+18.00%	+9.97%	+20.48%	+11.91%	+37.98%	+23.04%
ETTm1	96	0.300	0.355	0.290	0.351	0.300	0.355	**0.283**	**0.344**
	192	0.343	0.385	0.334	0.379	0.343	0.384	**0.330**	**0.374**
	336	0.393	0.416	0.374	0.406	0.390	0.414	**0.372**	**0.403**
	720	0.547	0.520	**0.478**	**0.480**	0.538	0.514	0.486	0.485
	Avg.	0.396	0.419	0.369	0.404	0.393	0.417	**0.368**	**0.402**
	Avg Imp.	–	–	+6.78%	+3.52%	+0.79%	+0.48%	+7.10%	+4.10%
ETTm2	96	0.263	0.340	**0.203**	**0.295**	0.260	0.339	0.205	0.295
	192	0.413	0.446	0.274	0.358	0.410	0.444	**0.247**	**0.334**
	336	0.617	0.551	0.352	0.407	0.611	0.550	**0.285**	**0.355**
	720	1.025	0.721	**0.505**	**0.491**	0.999	0.708	0.748	0.601
	Avg.	0.579	0.514	**0.334**	**0.388**	0.570	0.510	0.371	0.396
	Avg Imp.	–	–	+42.44%	+24.65%	+1.61%	+0.77%	+35.94%	+22.95%
Weather	96	0.148	0.216	0.145	0.210	0.147	0.215	**0.143**	**0.205**
	192	0.193	0.260	0.190	0.255	0.193	0.259	**0.187**	**0.250**
	336	0.250	0.313	0.244	0.304	0.249	0.312	**0.240**	**0.298**
	720	0.325	0.369	0.323	0.364	0.325	0.370	**0.302**	**0.341**
	Avg.	0.229	0.289	0.225	0.283	0.228	0.289	**0.218**	**0.274**
	Avg Imp.	–	–	+1.43%	+2.12%	+0.19%	+0.24%	+4.69%	+5.50%
Exchange	96	0.193	0.344	0.179	0.328	0.183	0.338	**0.141**	**0.282**
	192	1.015	0.783	0.934	0.742	0.910	0.739	**0.687**	**0.632**
	336	0.811	0.707	0.787	0.695	0.773	0.695	**0.578**	**0.602**
	720	1.146	0.882	1.137	0.878	1.151	0.889	**1.124**	**0.868**
	Avg.	0.791	0.679	0.759	0.661	0.754	0.665	**0.632**	**0.596**
	Avg Imp.	–	–	+4.05%	+2.69%	+4.64%	+2.02%	+20.05%	+12.25%
Illness	24	3.210	1.158	3.207	1.157	**2.833**	**1.108**	3.119	1.163
	36	3.386	1.188	3.382	1.187	**2.935**	**1.114**	3.382	1.191
	48	3.552	1.217	3.550	1.216	**3.220**	**1.170**	3.552	1.216
	60	3.692	1.250	3.689	1.249	**3.553**	**1.241**	3.692	1.250
	Avg.	3.460	1.203	3.457	1.202	**3.135**	**1.158**	3.436	1.205
	Avg Imp.	–	–	+0.09%	+0.05%	+9.39%	+3.70%	+0.69%	−0.17%
Solar	96	0.196	0.223	0.197	0.221	0.196	0.222	**0.193**	**0.212**
	192	0.200	0.231	0.196	0.231	0.200	0.231	**0.185**	**0.223**
	336	0.201	0.247	0.202	0.246	0.202	0.247	**0.192**	**0.235**
	720	0.745	0.683	0.746	0.696	0.745	0.684	**0.348**	**0.415**
	Avg.	0.336	0.346	0.335	0.349	0.336	0.346	**0.230**	**0.271**
	Avg Imp.	–	–	+0.24%	−0.72%	+0.05%	+0.02%	+31.63%	+21.60%

Table 7 provides the most heterogeneous comparison among the five forecasting backbones and makes clear that no single test-time mechanism dominates every setting. TTC achieves the best horizon-averaged MSE and MAE on ETTh2, ETTm1, Weather, Exchange, and Solar, while TAFAS is stronger on ETTh1 and ETTm2 and SOLID performs best on Illness. The magnitude of the differences also varies substantially across datasets, with large TTC gains in settings such as ETTh2 and Solar but much smaller or absent advantages in others. This behavior suggests that the usefulness of historical residual correction depends on the interaction between backbone behavior, data characteristics, prediction horizon, and the local residual structure available in memory. Accordingly, TTC should be interpreted as a parameter-update-free prediction-level alternative rather than a method expected to replace every adaptation strategy in every case. The combination of clear successes and meaningful exceptions supports a balanced view of TTC as broadly applicable but context dependent.

D Additional Ablation and Sensitivity Analysis

The main paper reports the complete residual-calibration strategy ablation and the cross-backbone sensitivity analyses for retrieval size K and correction strength λ. This appendix provides complementary analyses that are not expanded in the main text, including the effect of correction-ratio fallback, retrieval representation, horizon-wise sensitivity, softmax temperature, and memory stride.

Unless otherwise stated, the experiments in this section use Weather with $H = 96$ as the representative setting. Experiments involving multiple forecasting horizons use DLinear and evaluate $H \in \{96, 192, 336, 720\}$.

D.1 Effect of Correction-Ratio Fallback

The residual-strategy ablation in the main paper shows that multi-neighbor retrieval provides substantially more stable calibration than global, random, or single-neighbor residual reuse. We further isolate the role of the correction-ratio fallback after similarity-weighted Top-K aggregation.

Both variants in Table 8 use the same residual memory, retrieval representation, retrieved neighbors, similarity weights, and residual aggregation. The only difference is whether the resulting candidate correction is always applied or is rejected when its magnitude is too large relative to the Base prediction.

The unfiltered weighted correction already improves MSE for all five backbones, indicating that the retrieved residual neighborhood provides a useful calibration signal in this setting. However, applying every candidate correction is not always optimal. According to the unrounded values, introducing the correction-ratio fallback further improves both MSE and MAE for all five backbones.

The fallback remains selective rather than dominant. Only 1.2%–8.0% of variable-wise candidate corrections are rejected, while more than 92% are retained for every backbone. The gain therefore results from suppressing a relatively small subset of over-amplified corrections rather than from broadly disabling residual calibration. This supports the role of the correction-ratio rule as a conservative safeguard against potentially harmful over-correction.

D.2 Effect of Retrieval Representation

We first analyze the effect of the retrieval representation used to identify historically relevant residual trajectories. We compare three alternatives: V1 uses the raw look-back sequence, V2 combines window statistics with normalized temporal shape, and V3 further adds normalized first-order differences. Detailed definitions are provided in Appendix B.2.

The retrieval representation determines which historical residual trajectories are considered locally relevant to the current query. V1 operates directly on the raw look-back sequence before the final embedding normalization. V2 separates window-level statistics from the normalized temporal shape, while V3 additionally incorporates normalized first-order temporal changes. These variants progressively introduce information about local level, shape, and short-term dynamics without relying on backbone-specific hidden states.

As shown in Figure 5, the relative effectiveness of the representation variants differs across forecasting backbones. However, V3, which combines complementary statistical, shape, and difference information, provides the strongest average calibration performance across the evaluated models. This result motivates the use of V3 as the default retrieval representation in TTC.

D.3 Horizon-Wise Sensitivity to Retrieval Size and Correction Strength

We next examine two hyperparameters that directly control how retrieved historical residuals are used: the retrieval size K, which determines how many neighbors contribute to the residual estimate, and the correction strength λ, which controls the magnitude of the residual compensation added to the frozen Base prediction.

Retrieval size. The retrieval size K controls how many historical residual trajectories are aggregated to form the local compensation estimate. When K is very small, the correction depends

Table 8: Effect of correction-ratio fallback on Weather with $H = 96$. Unfiltered applies every similarity-weighted residual correction, whereas Ratio fallback preserves the Base prediction when the relative correction magnitude exceeds the predefined threshold. Acc. and Rej. denote the percentages of accepted and rejected variable-wise corrections. Lower MSE and MAE are better, and boldface indicates the better calibrated result for each backbone.

Policy	Backbone	Base		Calibrated		Decision Stat.	
		MSE	MAE	MSE	MAE	Acc.	Rej.
Unfiltered	iTransformer	0.175	0.214	0.165	0.209	100.00	0.00
	PatchTST	0.150	0.197	0.146	0.198	100.00	0.00
	DLinear	0.178	0.243	0.161	0.218	100.00	0.00
	Autoformer	0.269	0.343	0.251	0.328	100.00	0.00
	Crossformer	0.148	0.216	0.147	0.207	100.00	0.00
Ratio fallback	iTransformer	0.175	0.214	**0.163**	**0.206**	95.30	4.70
	PatchTST	0.150	0.197	**0.145**	**0.195**	96.30	3.70
	DLinear	0.178	0.243	**0.156**	**0.214**	94.80	5.20
	Autoformer	0.269	0.343	**0.251**	**0.327**	98.80	1.20
	Crossformer	0.148	0.216	**0.142**	**0.205**	92.00	8.00

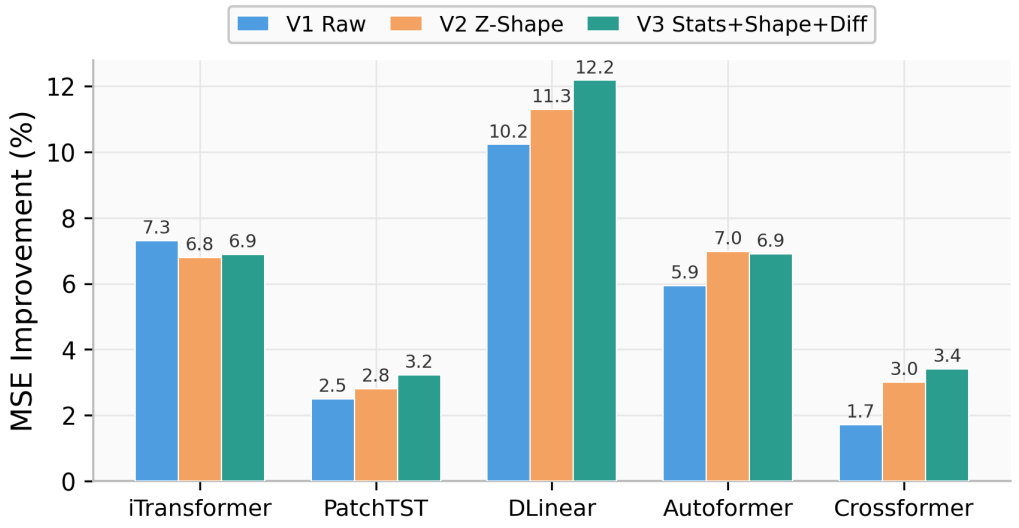
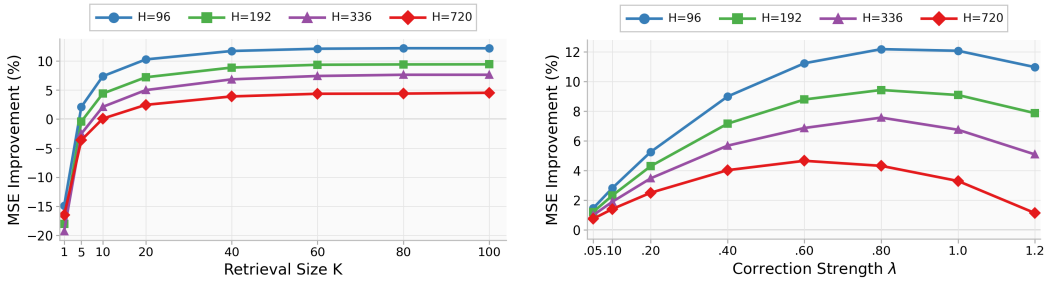


Figure 5: Effect of retrieval representation on Weather with $H = 96$. V1 uses the raw look-back sequence, V2 combines window statistics with normalized temporal shape, and V3 additionally includes normalized first-order differences.

heavily on a few individual residual trajectories and can become unstable, which is consistent with the single-neighbor ablation in the main paper. As shown in Fig. 6(a), increasing K produces a sharp improvement across all four horizons, after which the curves gradually saturate. The gains remain stable over the larger neighborhood sizes evaluated, indicating that multi-neighbor aggregation is the key stabilizing factor and that TTC is relatively insensitive once a sufficiently broad neighborhood is available.

Correction strength. The correction strength λ controls the magnitude of the retrieved residual compensation added to the frozen Base prediction. When λ is small, the retrieved residual contributes only a limited adjustment and the calibrated prediction remains close to the Base forecast. As shown in Fig. 6(b), increasing λ strengthens the compensation and consistently improves the result over the lower part of the sweep. Across the four horizons, the curves then enter a stronger intermediate-to-high region and eventually flatten or decline as the correction becomes overly aggressive. This pattern supports a bounded correction strength and indicates that TTC remains effective over a broad range rather than depending on a single isolated value.



(a) Retrieval size K

(b) Correction strength λ

Figure 6: Horizon-wise sensitivity to retrieval size K and correction strength λ on Weather using DLinear. The curves report MSE improvement over the frozen Base.

D.4 Sensitivity to Softmax Temperature and Memory Stride

We further examine the sensitivity of TTC to two practical hyperparameters: the softmax temperature τ, which controls how the retrieved Top-K neighbors are weighted during residual aggregation, and the memory stride, which controls how densely historical windows are retained in the residual memory.

Softmax temperature. A smaller temperature produces a more concentrated weighting distribution, causing the closest retrieved neighbors to dominate the aggregated correction. The results in Figure 7(a) show that very small temperatures can noticeably reduce the calibration gain for several backbones. As τ increases, performance improves rapidly and then enters a broad plateau, with only minor changes across the upper part of the evaluated range. This indicates that TTC benefits from sufficient multi-neighbor smoothing and is not highly sensitive once the temperature moves beyond the strongly concentrated regime.

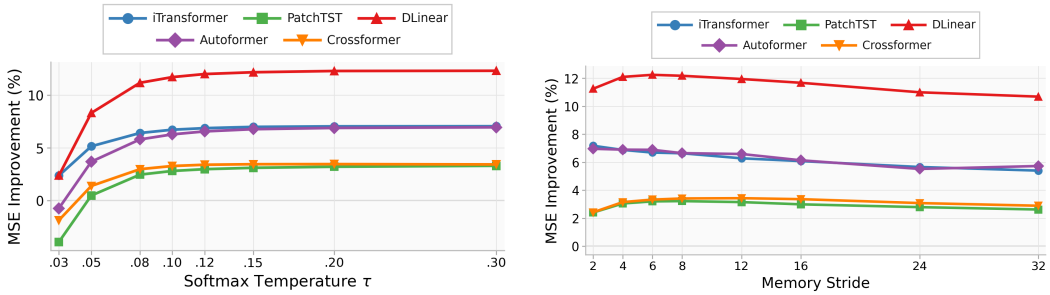
Memory stride. The memory stride determines how densely overlapping historical windows are stored in the residual memory and therefore directly controls the number of retrieval candidates. Figure 7(b) shows that small-to-moderate strides retain strong calibration performance, while the corresponding memory-bank size decreases sharply as the stride increases. At larger strides, the MSE improvement gradually declines for most backbones, but remains positive even with substantially fewer stored windows. This result exposes a practical coverage–efficiency trade-off: a denser memory provides stronger retrieval coverage, whereas a sparser memory can substantially reduce storage and exhaustive-search cost with a gradual rather than abrupt loss in calibration quality.

E Limitations and Future Directions

TTC provides a simple prediction-level calibration mechanism for frozen time-series forecasters, but several limitations remain and motivate further investigation.

First, TTC introduces an external historical memory and therefore incurs additional storage and retrieval cost. Under exhaustive search, the retrieval cost grows with the number of stored historical windows, the number of variables, and the dimensionality of the retrieval representation, as discussed in Appendix B. Memory stride can reduce this cost by removing redundant historical windows, but very large memories may still become expensive in high-dimensional or long-running deployment settings. Approximate nearest-neighbor indexing, memory compression, representative prototype selection, or hierarchical retrieval are promising directions for improving scalability without changing the downstream residual-calibration formulation.

Second, the current retrieval representation is intentionally lightweight and data-centric. It combines statistical summaries, normalized temporal shape, and first-order differences, which provides a model-independent representation that can be shared across heterogeneous forecasting backbones. However, this hand-crafted representation may not capture all forms of temporal similarity that are



(a) Softmax temperature τ

(b) Memory stride

Figure 7: Sensitivity to softmax temperature τ and memory stride on Weather with $H = 96$. Both panels report MSE improvement over the frozen Base across five backbones; the upper axis in panel (b) gives the approximate memory-bank size associated with each stride.

relevant to forecasting residuals. For example, longer-term periodic structure, multi-scale dynamics, regime information, or cross-variable interactions may provide additional retrieval signals. Future work could investigate learned or multi-scale retrieval representations while preserving the separation between the frozen forecasting backbone and the external calibration module.

Third, the final evaluation in this work uses a fixed historical residual memory during test-time prediction. This design provides a controlled evaluation protocol and avoids introducing additional online adaptation effects, but it does not exploit residual information that may become available later during deployment. In a streaming forecasting system, the future observations corresponding to previous predictions eventually become known and could be used to update, replace, or compress memory entries. Developing principled memory-update policies for such sequential settings is an important extension. In particular, future work could study how to balance recent and older residual evidence under gradual drift, abrupt regime changes, or recurring temporal patterns.

Fourth, the effectiveness of residual-memory calibration depends on the existence of historically relevant residual evidence. Similar observed contexts do not necessarily lead to identical future trajectories or forecasting errors, and some residual components may remain inherently irregular. Consequently, TTC should not be interpreted as guaranteeing an improvement for every individual prediction. The current correction-ratio fallback provides a conservative safeguard against excessively large candidate corrections, but it is not a formal estimator of retrieval uncertainty or correction reliability. More principled confidence estimation, neighbor-consistency modeling, or uncertainty-aware residual aggregation could provide stronger criteria for deciding when historical residual evidence should influence the Base forecast.

Finally, the present study focuses on deterministic point forecasting with multivariate long-term forecasting benchmarks. Historical residual memory may also be useful beyond point prediction, for example in probabilistic forecasting, prediction-interval calibration, anomaly-aware forecasting, or other sequential prediction problems in which historical prediction errors become observable over time. Extending TTC to these settings would require calibrating not only the prediction mean but also uncertainty or distributional properties, which we leave for future work.

Overall, the current results suggest that historical residuals can provide useful complementary information after a forecasting model has been trained and frozen. Future work will focus on making this residual evidence more scalable, adaptive, and uncertainty-aware while preserving the central deployment property of TTC: improving predictions without modifying the underlying forecasting backbone.