RETHINKING MULTIMODAL LEARNING FOR LARGE LANGUAGE MODELS IN TIME SERIES FORECASTING: THE PROBLEM OF PSEUDO-MULTIMODALITY

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-based time series forecasters often jointly model numerical histories and textual prompts generated from them. We note that these two representation pathways share the same sample observations and refer to this common-source, dual-view setting as pseudo-multimodality. In this setting, agreement between representations is insufficient to determine whether the language pathway provides additional predictive gains. We propose Conditional Incremental and Robust Fusion (CIRF), which uses the language pathway to conditionally correct a complete numerical forecast. CIRF learns to estimate the prompt representation from the numerical representation and decodes the residual between the observed and estimated prompt representations into a candidate correction. A gate supervised by forecasting-loss reduction learns whether to apply the correction for each sample and variable. CIRF-R further combines paired perturbation training with excess-sensitivity regularization to control the additional sensitivity introduced by corrections under shared input perturbations. Experiments on nine benchmarks demonstrate CIRF's strong forecasting performance against leading forecasting methods. Further experiments validate the effectiveness of selective correction and show that CIRF-R improves robustness under the evaluated input perturbations.

1 INTRODUCTION

Time series forecasting uses historical observations to anticipate future changes, supporting decisions in energy systems, environmental monitoring, and other domains where measurements evolve over time (Wu et al., 2021; Liu et al., 2025a). Progress in numerical forecasting has focused on representing trends, recurring patterns, and dependencies across time and variables so that these structures can inform predictions (Zeng et al., 2023; Nie et al., 2023; Liu et al., 2024a). Alongside these advances, researchers have explored adapting representations learned by pretrained large language models (LLMs) from extensive text corpora to time series forecasting (Jin et al., 2024). Prompt-based approaches connect language-model representations to time series through textual descriptions of the observed history and the forecasting task (Jin et al., 2024; Liu et al., 2025a). In designs that combine language and numerical pathways, a pretrained language model encodes these prompts into features that are used together with numerical representations (Liu et al., 2025a). This raises a central question: whether, and under what conditions, the language pathway improves prediction beyond the numerical pathway.

For sequence-derived prompts, the numerical and language pathways draw on a common observational source. In our setting, the numerical pathway encodes a historical window, while the language pathway encodes text describing historical statistics and trends from that window, together with fixed task and dataset information. Describing a rising trend in words changes its encoding and subsequent processing, but adds no observation of the current sample. This differs from supplying external news that can describe events absent from the numerical history (Wang et al., 2024). The system therefore has two representation pathways without acquiring a second source of sample-specific observations. We call this common-source setting *pseudo-multimodality*. In this setting, fusion combines different representations of the available observations, and the presence of a language pathway alone does not establish a complementary predictive benefit. The language representation may still improve the

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.

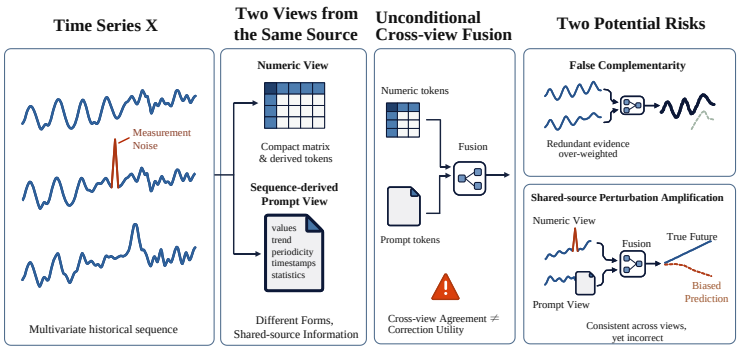


Figure 1: Pseudo-multimodality in sequence-derived LLM forecasting. Numerical and prompt representations are constructed from the same historical observations. Their agreement can reflect repeated evidence or a shared response to perturbations; what matters for forecasting is the conditional utility of the prompt-derived view beyond a complete numerical prediction.

numerical forecast, but its value for fusion must be assessed through predictive gains relative to the numerical pathway.

Formally, let $O = (X, M)$ denote the observations and metadata available for a sample. We study $Z_X = f(O)$ and $Z_P = g(T(O))$, where a template T constructs the prompt and neither pathway receives additional sample-specific observations beyond O. A learned numerical representation need not retain every predictive aspect of O, and a language encoder can transform the same evidence differently using its pretrained parameters. Thus, a shared source leaves room for complementary predictive utility relative to a particular numerical predictor. Figure 1 illustrates this common-source setting.

This distinction exposes an ambiguity in using representation relationships to guide forecasting. Similarity-based alignment can connect numerical and prompt embeddings (Liu et al., 2025a), but high agreement may reflect the same trend being encoded twice. Multimodal interactions can involve task-related redundancy, uniqueness, and synergy (Liang et al., 2023); agreement alone does not identify the prediction benefit of an additional view. Conversely, an improvement after adding a language pathway can arise from its architecture, optimization, or pretraining. Controlled replacements are needed to distinguish these explanations (Tan et al., 2024). We therefore study two linked questions: whether the language pathway improves the forecast for a given sample and variable, and whether the observed performance depends on language pretraining or sample-matched prompt content. The common observational source also matters when the history is corrupted. A perturbation can alter the numerical input and the prompt regenerated from it, exposing both pathways to the same disturbance. Their agreement can then reflect a shared response to corruption, and applying a correction can introduce output changes beyond those of the numerical branch. A derived-view correction therefore needs to be assessed both for its predictive utility and for the additional sensitivity it introduces under shared input perturbations.

To address these challenges, we propose Conditional Incremental and Robust Fusion (CIRF), which uses the derived language view to conditionally correct a complete numerical forecast. A learned predictor estimates the prompt representation from the numerical representation; subtracting this estimate yields a representation residual that is decoded into a candidate correction. A utility gate learns whether to apply the correction for each sample–variable pair, using the candidate's reduction in forecasting loss as supervision. At inference, the gate uses historical representations to select the correction or retain the numerical forecast. We further develop CIRF-R, which combines paired perturbation training with a penalty on prediction displacement beyond the numerical branch to control correction-induced sensitivity under shared input perturbations. Our contributions are:

- We characterize pseudo-multimodality in sequence-derived prompt forecasting, where numerical and language views share sample observations. We formulate a conditional-utility criterion that assesses the language view by its forecasting gains relative to a complete numerical forecast, addressing the ambiguity of inferring predictive complementarity from representational agreement.

- We propose CIRF, which separates the construction of language-derived corrections from the decision to apply them to a complete numerical forecast. It combines representation residuals with a utility gate supervised by forecasting-loss reduction to select corrections for each sample and variable. We further develop CIRF-R, combining paired perturbation training with excess-sensitivity regularization to control correction-induced sensitivity under shared input perturbations.
- Experiments on nine benchmarks show that CIRF achieves the lowest horizon-averaged MSE and MAE among the compared methods on four datasets. Fixed-model comparisons demonstrate the benefit of selective correction over mandatory application. CIRF-R further reduces relative MSE degradation compared with clean-trained CIRF under the tested ETTh1 perturbations.

2 Related Work

Numerical forecasters use decomposition, temporal patches, and variate-centric representations (Zeng et al., 2023; Nie et al., 2023; Liu et al., 2024a). LLM-based forecasting includes numerical reprogramming (Jin et al., 2024), autoregressive adaptation (Liu et al., 2024b), semantic-anchor retrieval (Pan et al., 2024), and cross-modal, multi-scale, or asynchronous interactions (Liu et al., 2025b; Jia et al., 2026; Zhang et al., 2026). TimeCAP uses LLM-generated summaries and retrieved examples for event prediction (Lee et al., 2025). Sequence-derived text reorganizes observed evidence, whereas external news can supply additional observations (Wang et al., 2024). Controlled LLM replacements can preserve forecasting performance in studied methods (Tan et al., 2024), motivating explicit tests of the contribution of language pretraining.

Multimodal and multi-view learning distinguish agreement from task-relevant complementarity (Baltrušaitis et al., 2019; Xu et al., 2013) and analyze redundancy, uniqueness, and synergy (Liang et al., 2023). Shared–private learning uses orthogonality constraints to separate representations (Bousmalis et al., 2016). Representation predictability and orthogonality do not establish uniquely informative content or forecasting utility. TimeCMA aligns numerical and prompt representations through similarity-based channel-wise aggregation (Liu et al., 2025a). T3Time combines horizon-aware temporal–frequency gating, adaptive cross-modal alignment, and channel-wise residual fusion before decoding (Chowdhury et al., 2026). CIRF instead constructs a residual relative to a learned numerical-to-prompt predictor, then uses forecasting-loss reduction to supervise binary acceptance of its decoded correction to a complete numerical forecast. This separates candidate construction from the decision to apply it.

Selective prediction learns when to abstain (Geifman & El-Yaniv, 2019); CIRF rejects an optional correction while retaining a complete forecast. Work on outlier impact (Li et al., 2021) and robust forecasting (Cheng et al., 2024) studies anomalous or contaminated observations. CIRF-R targets correction-induced sensitivity when a perturbation changes both the history and its regenerated prompt, penalizing output displacement beyond that of the numerical branch.

3 Method

3.1 Conditional Correction Formulation

Let $X_i = [x_{i,1}, \ldots, x_{i,C}] \in \mathbb{R}^{L\times C}$ and $Y_i \in \mathbb{R}^{H\times C}$ denote the historical input and forecasting target of sample i, respectively. In our setting, X_i is the only sample-dependent observation, and a deterministic template constructs $P_{i,c} = T(x_{i,c})$ from each variable history and fixed task information. Both pathways therefore share an observational source; their different representations may still have different predictive utility. CIRF makes this utility assessable by pairing a complete numerical default with an optional prompt-derived correction (Figure 2):

$$\hat{Y}_{i,c} = \hat{Y}_{X,i,c} + g_{i,c}\Delta\hat{Y}_{P,i,c}, \qquad g_{i,c} \in \{0, 1\}. \tag{1}$$

Here, $\hat{Y}_{X,i,c}, \Delta\hat{Y}_{P,i,c} \in \mathbb{R}^H$ denote the default forecast and candidate increment for variable c. The scalar $g_{i,c}$ accepts or rejects the entire H-step correction for each sample–variable pair. CIRF jointly learns the candidate correction and the decision to apply it.

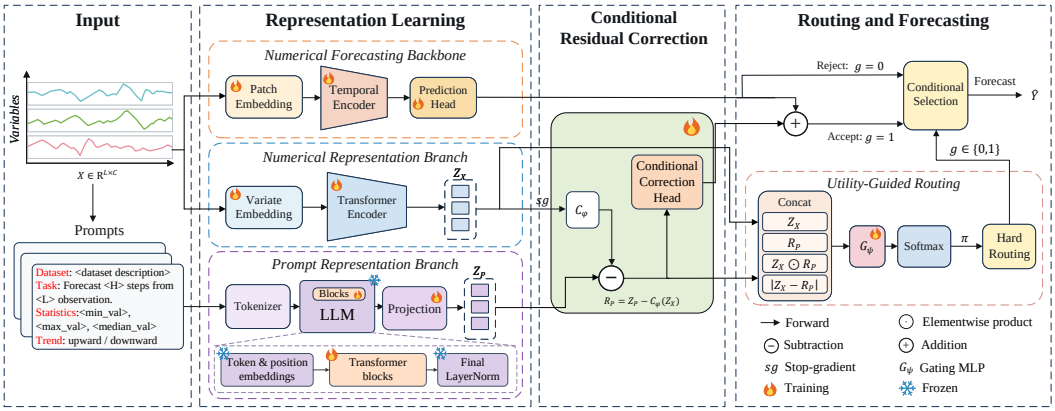


Figure 2: Overview of CIRF. The numerical forecasting backbone produces a default forecast, while the numerical and prompt representation branches encode the same historical input. A conditional predictor estimates the prompt representation from the numerical representation; subtracting this estimate yields a residual that is decoded into a candidate correction. A utility gate uses the numerical representation and conditional residual to select between the default forecast and the corrected candidate for each sample–variable pair.

3.2 Forecasting and Representation Learning

CIRF normalizes each history along time and restores the original scale after prediction; these transformations are implicit below. Additive forecasts share a common space, and each loss compares predictions and targets at the same scale.

Numerical Forecasting Backbone. Following PatchTST's channel-independent design (Nie et al., 2023), the backbone shares parameters across variables and processes each history $x_{i,c} \in \mathbb{R}^L$ separately. Repeating its final observation s times and extracting length-p patches with stride s gives $B_{i,c} \in \mathbb{R}^{N_p \times p}$, where $N_p = \lfloor (L-p)/s \rfloor + 2$. A projection $W_b \in \mathbb{R}^{p \times d_b}$ and learned positional embeddings $E_{\text{pos}} \in \mathbb{R}^{N_p \times d_b}$ produce tokens $V_{i,c} = B_{i,c} W_b + E_{\text{pos}} \in \mathbb{R}^{N_p \times d_b}$.

Pre-LN Transformer blocks produce $F_{i,c} = \text{Enc}_{\text{time}}(V_{i,c}) \in \mathbb{R}^{N_p \times d_b}$ using multi-head self-attention across temporal patches and token-wise feed-forward networks, with residual connections around both sublayers. Flattening and projecting the encoded features yields the full-horizon forecast:

$$\hat{Y}_{X,i,c} = W_o \,\text{Flatten}(F_{i,c}) + b_o, \tag{2}$$

where Flatten concatenates its rows into an $N_p d_b$-dimensional column vector. The weights $W_o \in \mathbb{R}^{H \times N_p d_b}$ and bias $b_o \in \mathbb{R}^H$ produce $\hat{Y}_{X,i,c} \in \mathbb{R}^H$; stacking forecasts as columns gives $\hat{Y}_{X,i} \in \mathbb{R}^{H \times C}$. Training dropout in the encoder and before projection is omitted from the notation. The backbone has separate parameters from the numerical representation branch and supplies the default forecast.

Numerical Representation Branch. The numerical encoder embeds each variable's complete history as one token, $H_i^{(0)} = \text{Linear}(X_i^\top) \in \mathbb{R}^{C \times d}$, using a shared linear projection from L observations to d features. The tokens pass through K Pre-LN Transformer blocks (Vaswani et al., 2017; Xiong et al., 2020) to capture dependencies across variables. Block $\ell = 0, \ldots, K-1$ computes

$$\begin{aligned} U_i^{(\ell)} &= H_i^{(\ell)} + \text{MHSA}_\ell\Big(\text{LN}_{\ell,1}(H_i^{(\ell)})\Big), \\ H_i^{(\ell+1)} &= U_i^{(\ell)} + \text{FFN}_\ell\Big(\text{LN}_{\ell,2}(U_i^{(\ell)})\Big), \end{aligned} \tag{3}$$

where $H_i^{(\ell)}, U_i^{(\ell)} \in \mathbb{R}^{C \times d}$ are the block input and intermediate state. Layer normalization acts along features; multi-head self-attention (MHSA) spans the C variable tokens ($C \times C$ per head), and FFN acts token-wise with shared parameters. The output $Z_{X,i} = H_i^{(K)} \in \mathbb{R}^{C \times d}$ supplies $z_{X,i,c} \in \mathbb{R}^d$ for residual construction and routing; dropout is omitted.

Prompt Representation Branch. For each variable, template T forms $P_{i,c}$ from a fixed dataset description, input and prediction lengths, and the normalized history's minimum, maximum, median, and trend direction. The language model's tokenizer converts the prompt to tokens, whose embeddings are combined with positional embeddings to form $H^{(0)}_{P,i,c}$. A pretrained language model encodes the prompt with K_P Pre-LN Transformer blocks. For $\ell = 0, \ldots, K_P - 1$,

$$\begin{aligned} U^{(\ell)}_{P,i,c} &= H^{(\ell)}_{P,i,c} + \mathrm{MMSA}_\ell\Big(\mathrm{LN}^P_{\ell,1}(H^{(\ell)}_{P,i,c})\Big), \\ H^{(\ell+1)}_{P,i,c} &= U^{(\ell)}_{P,i,c} + \mathrm{FFN}^P_\ell\Big(\mathrm{LN}^P_{\ell,2}(U^{(\ell)}_{P,i,c})\Big), \end{aligned} \tag{4}$$

where $H^{(\ell)}_{P,i,c}, U^{(\ell)}_{P,i,c} \in \mathbb{R}^{n_{i,c} \times d_{\mathrm{LM}}}$ are the block input and intermediate state, with $n_{i,c}$ valid tokens after truncation and hidden size d_{LM}. Masked multi-head self-attention (MMSA) is causal: each token attends to itself and preceding tokens. Layer normalization acts along features, and FFN^P acts token-wise. Padding and dropout are omitted.

After applying the final layer normalization $\mathrm{LN}_{\mathrm{out}}$, we take the last non-padding token state $h_{P,i,c} \in \mathbb{R}^{d_{\mathrm{LM}}}$ as the prompt summary. We obtain $z_{P,i,c} = \mathrm{LN}_P(W_P h_{P,i,c} + b_P)$ in the d-dimensional decision space, where $W_P \in \mathbb{R}^{d \times d_{\mathrm{LM}}}$ and $b_P \in \mathbb{R}^d$ are trainable projection parameters, and LN_P normalizes the projected features. Stacking $z_{P,i,c} \in \mathbb{R}^d$ as rows gives $Z_{P,i} \in \mathbb{R}^{C \times d}$, paired by variable with $Z_{X,i}$. Sample-dependent prompt content comes entirely from X_i; pretraining supplies an inductive prior.

3.3 Conditional Residual Correction

CIRF uses the discrepancy between the prompt representation and a learned prediction from the numerical representation to construct its candidate correction. Predictable overlap guides candidate construction; forecasting utility determines whether to apply it.

Conditional Residual Construction. A conditional predictor C_ϕ, shared across variables, maps $z_{X,i,c}$ to an estimate of $z_{P,i,c}$. Its prediction and the resulting conditional residual are

$$S_{P,i,c} = C_\phi\big(\mathrm{sg}(z_{X,i,c})\big), \qquad R_{P,i,c} = z_{P,i,c} - S_{P,i,c}, \tag{5}$$

where $C_\phi : \mathbb{R}^d \to \mathbb{R}^d$, and $\mathrm{sg}(\cdot)$ denotes stop-gradient. The vectors $S_{P,i,c}, R_{P,i,c} \in \mathbb{R}^d$ are the predicted component and its residual, respectively. Stacking the residuals gives $R_{P,i} \in \mathbb{R}^{C \times d}$. This residual is defined relative to the learned numerical representation used for correction decisions and the conditional predictor. To learn the conditional mapping, we minimize a reconstruction objective with both representations detached:

$$\mathcal{L}_{\mathrm{red}} = \mathbb{E}_{i,c}\left[\frac{1}{d}\left\|\mathrm{sg}(z_{P,i,c}) - C_\phi\big(\mathrm{sg}(z_{X,i,c})\big)\right\|_2^2\right]. \tag{6}$$

The stop-gradients restrict updates from $\mathcal{L}_{\mathrm{red}}$ to C_ϕ. Other objectives train the encoders, and downstream gradients through $R_{P,i,c}$ also update C_ϕ.

Residual Regularization. To discourage directional overlap between the residual and numerical representation, we use $\mathcal{L}_{\mathrm{orth}} = \mathbb{E}_{i,c}\big[\cos^2(z_{X,i,c}, R_{P,i,c})\big]$, where $\cos(\cdot,\cdot)$ denotes cosine similarity. Squaring penalizes parallel and antiparallel directions independently of magnitude. This soft geometric constraint complements reconstruction without imposing statistical independence.

Conditional Correction Head. The correction head D_R affinely projects the residual to a forecast increment. Adding it to the numerical default gives the candidate forecast:

$$\Delta\hat{Y}_{P,i,c} = W_R R_{P,i,c} + b_R, \qquad \hat{Y}^+_{i,c} = \hat{Y}_{X,i,c} + \Delta\hat{Y}_{P,i,c}, \tag{7}$$

where $W_R \in \mathbb{R}^{H \times d}$ and $b_R \in \mathbb{R}^H$ are the projection parameters of D_R, and $\Delta\hat{Y}_{P,i,c}, \hat{Y}^+_{i,c} \in \mathbb{R}^H$ are the full-horizon increment and candidate forecast. Direct candidate supervision links the residual to forecasting; its loss relative to the numerical default supplies the gate's utility target.

3.4 UTILITY-GUIDED ROUTING

Routing Decision. For each sample–variable pair, the gate predicts correction utility from the numerical representation, conditional residual, their elementwise product, and absolute difference. A learned gate $G_\psi : \mathbb{R}^{4d} \to \mathbb{R}^2$ maps these features to rejection and acceptance logits:

$$\begin{aligned} u_{i,c} &= \big[z_{X,i,c}; R_{P,i,c}; z_{X,i,c} \odot R_{P,i,c}; |z_{X,i,c} - R_{P,i,c}|\big], \\ \boldsymbol{s}_{i,c} &= G_\psi(u_{i,c}), \end{aligned} \tag{8}$$

where $u_{i,c} \in \mathbb{R}^{4d}$ concatenates the four feature vectors, and $\boldsymbol{s}_{i,c} = [s_{i,c,0}, s_{i,c,1}]^\top$ contains the rejection and acceptance logits, respectively. Implementation details for C_ϕ and G_ψ are given in Appendix A.1. Softmax converts the logits to routing probabilities $\boldsymbol{\pi}_{i,c} = \mathrm{Softmax}(\boldsymbol{s}_{i,c})$. During training, we use a straight-through estimator (Bengio et al., 2013) for the binary gate:

$$g_{i,c} = \mathrm{sg}(\mathbb{I}[\pi_{i,c,1} > \tau] - \pi_{i,c,1}) + \pi_{i,c,1}, \tag{9}$$

where $\boldsymbol{\pi}_{i,c} = [\pi_{i,c,0}, \pi_{i,c,1}]^\top$ contains rejection and acceptance probabilities, and $\tau \in [0, 1]$ is the threshold. The forward value is $\mathbb{I}[\pi_{i,c,1} > \tau]$, while backpropagation follows $\pi_{i,c,1}$. Equation 1 therefore selects either the complete numerical forecast or the corrected candidate.

Utility Supervision. The gate is supervised by the candidate's loss reduction relative to the numerical default. Let ℓ denote the Huber loss (Huber, 1964) averaged over the H prediction steps. The binary utility target is

$$b^*_{i,c} = \mathbb{I}\big[\ell(\hat{Y}_{X,i,c}, Y_{i,c}) - \ell(\hat{Y}^+_{i,c}, Y_{i,c}) > \delta\big], \tag{10}$$

where $\delta \geq 0$ specifies the required loss reduction for a positive training label. The gate learns to estimate the probability of this event from $u_{i,c}$, with $\pi_{i,c,1}$ supervised by binary cross-entropy $\mathcal{L}_{\mathrm{gate}} = -\mathbb{E}_{i,c}\big[b^*_{i,c} \log \pi_{i,c,1} + (1 - b^*_{i,c}) \log \pi_{i,c,0}\big]$. The margin δ defines the training labels, whereas τ controls acceptance from the predicted probability. Future targets are used only to construct training supervision. At inference, routing uses historical-window representations and the fixed threshold τ.

3.5 TRAINING OBJECTIVES

Forecasting objective. With ℓ_{H} denoting Huber loss, CIRF supervises the routed prediction, the complete numerical default, and the candidate forecast:

$$\mathcal{L}_{\mathrm{fore}} = \ell_{\mathrm{H}}(Y, \hat{Y}) + \lambda_b \ell_{\mathrm{H}}(Y, \hat{Y}_X) + \lambda_c \ell_{\mathrm{H}}(Y, \hat{Y}^+). \tag{11}$$

The three terms train the final prediction, preserve the numerical default branch, and directly supervise the candidate correction, respectively.

Sensitivity regularization. To control correction-induced sensitivity, we sample a perturbation operator A to obtain $\widetilde{X} = A(X)$ and regenerate the prompt from the perturbed history. The clean and perturbed passes share parameters and targets. Their representation changes define a common-source risk weight $q_{i,c}$ from the relative changes and directional agreement of numerical and prompt representations. The resulting sensitivity loss is

$$\mathcal{L}_{\mathrm{sens}} = \mathbb{E}_{i,c}\left[q_{i,c}[e_{i,c} - e^X_{i,c}]_+\right], \tag{12}$$

where $e_{i,c}$ and $e^X_{i,c}$ measure the perturbed displacement of the selected prediction and numerical default, respectively. This term penalizes sensitivity introduced by the correction path beyond the numerical forecast.

Joint optimization. For standard CIRF, clean objectives are optimized without the sensitivity term. CIRF-R additionally enables the sensitivity-aware objective:

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{fore}} + \lambda_r \mathcal{L}_{\mathrm{red}} + \beta \mathcal{L}_{\mathrm{orth}} + \lambda_g \mathcal{L}_{\mathrm{gate}} + \lambda_a \mathcal{L}_{\mathrm{sens}}. \tag{13}$$

Setting $\lambda_a = 0$ gives standard CIRF, whereas $\lambda_a > 0$ enables sensitivity-aware training. Detailed risk construction and optimization procedures are provided in Appendices A and B.

Table 1: Long-term forecasting performance averaged across four prediction horizons. MSE and MAE are reported; lower values are better. Bold and underlined values denote the best and second-best results, respectively.

Dataset	DLinear		TimesNet		PatchTST		Time-LLM		TimeCMA		SE-LLM		CIRF	
	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓
ETTh1	0.456	0.452	0.458	0.450	0.469	0.455	<u>0.448</u>	0.443	0.466	0.459	**0.439**	**0.437**	0.459	<u>0.442</u>
ETTh2	0.559	0.515	0.414	0.427	0.387	0.407	<u>0.381</u>	<u>0.404</u>	0.404	0.420	0.395	0.412	**0.368**	**0.394**
ETTm1	0.403	0.407	0.400	0.406	**0.387**	<u>0.400</u>	0.410	0.409	0.407	0.416	0.433	0.426	<u>0.395</u>	**0.396**
ETTm2	0.350	0.401	0.291	0.333	**0.281**	**0.326**	0.296	0.341	0.293	0.336	<u>0.285</u>	0.328	**0.281**	<u>0.327</u>
Weather	0.265	0.317	<u>0.259</u>	0.286	<u>0.259</u>	<u>0.280</u>	0.275	0.291	<u>0.259</u>	0.285	0.261	<u>0.280</u>	**0.255**	**0.275**
Electricity	0.212	0.300	<u>0.193</u>	0.295	0.205	0.290	0.196	<u>0.288</u>	0.249	0.346	0.221	0.305	**0.190**	**0.275**
Solar-Energy	0.330	0.401	0.301	0.319	<u>0.270</u>	<u>0.307</u>	0.405	0.407	0.347	0.384	0.282	0.308	**0.234**	**0.269**
Exchange	**0.354**	0.414	0.416	0.443	0.366	0.404	<u>0.356</u>	0.403	0.404	0.433	0.360	<u>0.402</u>	0.359	**0.399**
ILI	2.615	1.090	**2.139**	<u>0.924</u>	2.388	1.011	2.432	1.012	4.397	1.521	2.467	0.977	<u>2.210</u>	**0.917**

4 EXPERIMENTS

Datasets and setup. We evaluate CIRF on nine public benchmarks: ETTh1, ETTh2, ETTm1, ETTm2, Weather, Electricity, Solar-Energy, Exchange, and ILI. Following standard long-term forecasting protocols (Wu et al., 2021; Nie et al., 2023), we use input length 96 and horizons $H \in \{96, 192, 336, 720\}$, except for ILI (input length 36; $H \in \{24, 36, 48, 60\}$). Data are split chronologically into training, validation, and test sets, with normalization statistics computed only from the training split. We report MSE and MAE.

Baselines. Numerical baselines are DLinear, TimesNet (Wu et al., 2023), and PatchTST; LLM-based baselines are Time-LLM, TimeCMA, and SE-LLM (Liu et al., 2026). Results combine local runs with published results matching the stated protocol.

Implementation details. CIRF uses a PatchTST-style numerical backbone and the first six blocks of GPT-2 (Radford et al., 2019). AdamW jointly optimizes the numerical forecaster, CIRF task layers, and LLM blocks at separate learning rates. Controlled experiments use seeds 2021, 2022, and 2023, with seed 2021 shown in Table 3. For benchmark and routing runs, validation MSE selects the deployment threshold after checkpoint selection; it remains fixed during testing. Appendix B provides dataset details, result sources, and experiment-specific configurations.

4.1 FORECASTING ACCURACY ACROSS BENCHMARKS

Table 1 shows that CIRF achieves the lowest horizon-averaged MSE and MAE on ETTh2, Weather, Electricity, and Solar-Energy. The improvement is most pronounced on Solar-Energy, where CIRF reduces MSE/MAE from PatchTST's 0.270/0.307 to 0.234/0.269. On ETTh2, CIRF achieves 0.368/0.394, compared with Time-LLM's 0.381/0.404. Improvements over the best compared baselines on Weather and Electricity are smaller. CIRF also obtains lower MSE and MAE than TimeCMA on all nine benchmarks. The rankings differ across error metrics: on Exchange and ILI, CIRF attains the lowest MAE, whereas DLinear and TimesNet attain lower MSE, respectively. CIRF is competitive with both numerical and LLM-based forecasting methods, with its relative advantage varying across datasets and error metrics. Appendix C.1 provides the complete results at each horizon (Table 9).

4.2 REJECTABLE VERSUS MANDATORY CORRECTION

To test whether candidate corrections should always alter the numerical forecast, we compare four routing choices on ETTh1, ETTm2, and Weather at horizons 96 and 336. Each trained checkpoint supplies fixed numerical predictions and candidate corrections: numerical-only retains the default, unconditional routing applies every correction, and CIRF uses a validation-selected threshold. An oracle selects the lower-error branch for each sample–variable pair using future targets, providing a reference for selection potential. Table 2 reports these predictions and the learned gate's activation rate.

Table 2: Conditional correction routing results with oracle reference averaged over seeds 2021, 2022, and 2023. Activation denotes the percentage of accepted corrections.

Dataset	H	Numerical		Uncond.		CIRF		Oracle		Activation
		MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	(%)
ETTh1	96	0.378	0.392	0.383	0.398	0.379	0.393	0.371	0.389	4.136
ETTh1	336	0.491	0.453	0.503	0.460	0.491	0.453	0.486	0.450	1.499
ETTm2	96	0.175	0.257	0.176	0.259	0.175	0.257	0.170	0.253	7.877
ETTm2	336	0.303	0.342	0.309	0.347	0.303	0.342	0.288	0.332	0.210
Weather	96	0.176	0.213	0.173	0.211	0.172	0.210	0.155	0.194	77.508
Weather	336	0.279	0.294	0.278	0.296	0.277	0.294	0.263	0.284	77.173

Table 3: Ablations on ETTm2 at horizon 96. Full CIRF results are bold. Values are rounded to three decimal places; comparisons use unrounded results.

Metric	Full	w/o CI	w/o Rec.	w/o Orth.	Soft	Always
MSE↓	**0.176**	0.177	0.178	0.176	0.176	0.176
MAE↓	**0.258**	0.259	0.258	0.258	0.259	0.259

Selective CIRF achieves lower MSE and MAE than unconditional correction in all six settings. With both predictions fixed, the difference comes from choosing which corrections to apply. On ETT, unconditional correction raises both errors relative to the numerical branch. CIRF accepts fewer than 8% of corrections and remains close to the numerical forecast, with matching or slightly higher errors. On Weather, it accepts approximately 77% and lowers MSE relative to both branches. Conditional utility thus varies across the evaluated tasks even though their two views share the same observational source. The oracle's MSE is lower than the numerical branch's on ETT, revealing useful CIRF-generated candidates even where applying all corrections degrades forecasting. Its gap to CIRF separates the candidates' potential from the utility achieved by learned selection.

4.3 Ablating the Correction Decision

Table 3 presents an illustrative ETTm2 breakdown at horizon 96 for seed 2021, selected after inspecting the three-seed results. Variants are trained independently under the full model's training budget and validation protocol. Removing conditional incremental decomposition (CI) feeds the prompt representation directly to the correction branch. The reconstruction (Rec.) and orthogonality (Orth.) ablations set their respective loss weights to zero. Soft routing retains continuous probabilities; always-fuse fixes the gate to one.

Full CIRF achieves the best MAE in this setting. Feeding the prompt representation directly to the correction branch increases MSE and MAE by 0.631% and 0.259%, respectively. Removing reconstruction supervision increases MSE by 1.285%, leaving the predictor to learn through the remaining objectives. These comparisons favor a residual candidate trained with an explicit conditional prediction target. Hard routing also outperforms soft routing and always-fuse in both metrics: the gate can retain the numerical forecast instead of scaling or applying every increment.

4.4 Controlling Correction-Induced Sensitivity

A common observational source also exposes both pathways to the same corruption. We test this consequence on ETTh1 at horizons 96 and 336, applying Gaussian noise, impulse noise, or segment bias to normalized histories at $s \in \{0.05, 0.10, 0.20\}$ and regenerating prompts while leaving future targets unchanged (Appendix B.4). We compare the numerical backbone, clean-trained CIRF, and CIRF-R, which retains the architecture and standard optimization settings but adds paired training with the corresponding perturbation at $s = 0.10$ and sensitivity regularization. Relative degradation is $\mathrm{rMSE} = 100(\mathrm{MSE}_{\mathrm{noisy}} - \mathrm{MSE}_{\mathrm{clean}})/\mathrm{MSE}_{\mathrm{clean}}$, with rMAE defined analogously. Extra amplification (EA) measures correction-induced output displacement beyond the numerical branch. Table 4 averages the $s = 0.20$ results over both horizons and three seeds. CIRF-R lowers relative MSE degradation under all three perturbations compared with clean-trained CIRF. Mean rMSE falls from 46.873% to 33.690%, and mean EA from 0.004 to 0.001. Impulse noise produces the

Table 4: Robustness under common-source perturbations at $s = 0.20$. rMSE reports relative MSE degradation (%) and EA denotes extra amplification introduced by the correction pathway. A dash (–) indicates that EA is not applicable to the numerical-only backbone. Bold and underlined values denote the best and second-best results.

Model	Clean		rMSE↓ (%)				Mean EA↓
	MSE↓	MAE↓	Gauss.	Imp.	Seg.-bias	Mean	
Num. backbone	**0.446**	0.435	2.382	137.408	1.184	46.992	–
CIRF	0.446	0.435	2.366	136.912	1.340	46.873	0.004
CIRF-R	0.449	**0.434**	**1.837**	**98.729**	**0.505**	**33.690**	**0.001**

Table 5: Pretraining and architectural controls. PT and PM denote pretrained weights and sample-matched prompts. ΔMSE is relative to the numerical-only baseline. Bold and underlined values denote the best and second-best forecasting results, respectively.

Control	Configuration		Parameters (M)		Forecasting		
	PT	PM	Total	Trainable	MSE↓	MAE↓	ΔMSE (%)
CIRF	Yes	Yes	83.115	43.729	0.344	0.367	+0.913
Rand. GPT-2	No	Yes	83.115	43.729	0.344	0.368	+1.012
LLM2Attn	No	Yes	42.162	42.162	0.344	0.367	+1.013
LLM2Trsf	No	Yes	46.887	46.887	0.346	0.369	+1.548
P-mask	Yes	No	83.115	43.729	0.344	0.366	+0.937
P-shuffle	Yes	No	83.115	43.729	0.342	**0.366**	+0.305
Num.-only	N/A	N/A	0.878	0.878	**0.341**	0.367	**0.000**

largest reduction, from 136.912% to 98.729% rMSE. Clean MSE changes from 0.446 to 0.449 and MAE from 0.435 to 0.434. The combined perturbation-training and regularization scheme therefore reduces the extra sensitivity of correction under these shared input perturbations.

4.5 Pretraining and Architectural Controls

Following Tan et al. (2024), we independently retrain the controls in Table 5 on ETTh1 and ETTm2 at horizons 96 and 336 under a matched training protocol. Random GPT-2 and the attention replacement achieve errors close to CIRF, while masking or shuffling prompts produces no clear aggregate loss. Numerical-only prediction attains the lowest MSE, 0.341 versus 0.344 for CIRF. These retrained controls show no clear aggregate advantage from language pretraining or sample-matched prompt content in the evaluated ETT setting. Table 2 complements this finding: with trained candidates and numerical defaults fixed, selective routing improves over mandatory correction. These results show that selecting corrections can improve a trained model, but this improvement alone does not establish a benefit from language pretraining. Additional Weather attribution and cost controls are reported in Appendix D (Tables 10 and 11).

5 Conclusion

Sequence-derived prompts and numerical forecasts use the same observations, but their representations may differ in predictive utility. CIRF constructs prompt-derived candidate corrections to a complete numerical forecast and learns when to apply them; CIRF-R also targets correction-induced sensitivity. Across six fixed-model comparisons, selective routing improves on mandatory correction, while remaining close to or slightly worse than the numerical baseline on ETT and lowering MSE on Weather. Separately retrained ETT controls show no clear aggregate advantage from language pretraining or sample-matched prompts under the tested protocol. CIRF-R reduces correction-induced sensitivity under the evaluated ETTh1 perturbations. These results support assessing pseudo-multimodal fusion by the conditional predictive benefit of a derived view, rather than representational agreement alone.

REFERENCES

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2): 423–443, 2019. doi: 10.1109/TPAMI.2018.2798607.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. URL https://arxiv.org/abs/1308.3432.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *Advances in Neural Information Processing Systems*, volume 29, pp. 343–351, 2016.

Hao Cheng, Qingsong Wen, Yang Liu, and Liang Sun. RobustTSF: Towards theory and design of robust time series forecasting with anomalies. In *International Conference on Learning Representations*, 2024.

Abdul Monaf Chowdhury, Rabeya Akter, and Safaeid Hossain Arib. T3Time: Tri-modal time series forecasting via adaptive multi-head alignment and residual fusion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20597–20605, 2026. doi: 10.1609/aaai.v40i25.39196.

Yonatan Geifman and Ran El-Yaniv. SelectiveNet: A deep neural network with an integrated reject option. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2151–2159, 2019.

Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964. doi: 10.1214/aoms/1177703732.

Shuning Jia, Baijun Song, Canming Ye, and Chun Yuan. M3Time: LLM-enhanced multi-modal, multi-scale, and multi-frequency multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22265–22273, 2026. doi: 10.1609/aaai.v40i27.39383.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations*, 2024.

Geon Lee, Wenchao Yu, Kijung Shin, Wei Cheng, and Haifeng Chen. TimeCAP: Learning to contextualize, augment, and predict time series events with large language model agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18082–18090, 2025. doi: 10.1609/aaai.v39i17.33989.

Jianbo Li, Lecheng Zheng, Yada Zhu, and Jingrui He. Outlier impact characterization for time series data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11595–11603, 2021. doi: 10.1609/aaai.v35i13.17379.

Paul Pu Liang, Yun Cheng, Xiang Fan, Chun Kai Ling, Suzanne Nie, Richard J. Chen, Zihao Deng, Nicholas Allen, Randy Auerbach, Faisal Mahmood, Ruslan Salakhutdinov, and Louis-Philippe Morency. Quantifying and modeling multimodal interactions: An information decomposition framework. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. TimeCMA: Towards LLM-empowered multivariate time series forecasting via cross-modality alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18780–18788, 2025a. doi: 10.1609/aaai.v39i18.34067.

Hao Liu, Xiaoxing Zhang, Chun Yang, and Xiaobin Zhu. Semantic-enhanced time-series forecasting via large language models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a5e2853be2ff6218a8d87a5ce4b65062-Abstract-Conference.html.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. CALF: Aligning LLMs for time series forecasting via cross-modal fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18915–18923, 2025b. doi: 10.1609/AAAI.V39I18.34082.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted Transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024a.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. AutoTimes: Autoregressive time series forecasters via large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024b.

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.

Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. S^2IP-LLM: Semantic space informed prompt learning with LLM for time series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 39135–39153, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Mingtian Tan, Mike A. Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are language models actually useful for time series forecasting? In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1922.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Xinlei Wang, Maike Feng, Jing Qiu, Jinjin Gu, and Junhua Zhao. From news to forecast: Integrating event analysis in LLM-based time series forecasting with reflection. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1853.

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*, volume 34, pp. 22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the Transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10524–10533, 2020.

Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013.

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.

Fan Zhang, Shiming Fan, and Hua Wang. TimeSAF: Towards LLM-guided semantic asynchronous fusion for time series forecasting. *arXiv preprint arXiv:2604.12648*, 2026.

AI USE STATEMENT

We used AI tools to assist with drafting and revising portions of the manuscript, organizing the presentation, and polishing the language; searching for relevant literature and verifying sources; and discussing research ideas, providing feedback on methodology and experimental design, and interpreting experimental results. The authors reviewed the AI-assisted material and take full responsibility for the final manuscript, including its research ideas, methodology, experiments, analyses, conclusions, and references.

APPENDICES

A METHOD AND TRAINING DETAILS

A.1 CONDITIONAL PREDICTOR AND UTILITY GATE

The feed-forward networks in all three branches and the conditional predictor C_ϕ each use two affine layers separated by GELU. The predictor shares parameters across variables.

The two-layer utility gate computes the logits in Equation 8 as

$$G_\psi(u_{i,c}) = W_{g,2}\,\mathrm{GELU}(W_{g,1}u_{i,c} + b_{g,1}) + b_{g,2}.$$

For a gate hidden width d_g, the weights are $W_{g,1} \in \mathbb{R}^{d_g \times 4d}$ and $W_{g,2} \in \mathbb{R}^{2\times d_g}$, with biases $b_{g,1} \in \mathbb{R}^{d_g}$ and $b_{g,2} \in \mathbb{R}^2$. Dropout is applied after GELU during training and omitted from the notation.

A.2 COMMON-SOURCE SENSITIVITY REGULARIZATION

CIRF-R controls correction-induced sensitivity when both representations inherit perturbations from the same history. For a sampled perturbation operator A, we regenerate the prompt as

$$\widetilde{X} = A(X), \qquad \widetilde{P}_{i,c} = T(\widetilde{x}_{i,c}). \tag{14}$$

The clean and perturbed passes share parameters and targets. Let

$$d^X_{i,c} = \widetilde{z}_{X,i,c} - z_{X,i,c}, \qquad d^P_{i,c} = \widetilde{z}_{P,i,c} - z_{P,i,c}.$$

Their relative changes and directional agreement define the common-source risk weight:

$$q_{i,c} = \left[\cos(d^X_{i,c}, d^P_{i,c})\right]_+ \sqrt{\rho^X_{i,c}\rho^P_{i,c} + \epsilon}, \tag{15}$$

where

$$\rho^X_{i,c} = \frac{\|d^X_{i,c}\|_2}{\|z_{X,i,c}\|_2 + \epsilon}, \qquad \rho^P_{i,c} = \frac{\|d^P_{i,c}\|_2}{\|z_{P,i,c}\|_2 + \epsilon}.$$

The risk weight is used only for sensitivity regularization and is not provided to the utility gate. With $m_H(U,V) = H^{-1}\|U - V\|_1$, we define

$$e_{i,c} = m_H(\hat{Y}_{i,c}, \widetilde{\hat{Y}}_{i,c}), \qquad e^X_{i,c} = m_H(\hat{Y}_{X,i,c}, \widetilde{\hat{Y}}_{X,i,c}),$$

which enter the excess-sensitivity loss in Equation 12. It penalizes correction-induced sensitivity beyond the numerical forecast, rather than absolute displacement.

Training objective. The forecasting supervision in Equation 11 combines with reconstruction, orthogonality, and utility-gate losses in Equation 13. Standard CIRF sets $\lambda_a = 0$; CIRF-R enables $\lambda_a > 0$ using paired clean and perturbed histories. Experiment configurations and perturbations are specified in Appendix B.

B EXPERIMENTAL SETUP

B.1 DATASET DETAILS

We evaluate multivariate forecasting on nine public benchmarks. Table 6 summarizes their variables, sampling frequencies, observation counts, and domains. The ETT, Weather, Electricity, Exchange, and ILI benchmarks are used in prior forecasting studies (Wu et al., 2021; Nie et al., 2023).

ETT. The Electricity Transformer Temperature (ETT) datasets record six power-load variables and oil temperature from two electricity transformers.[1] ETTh1 and ETTh2 provide hourly measurements, while ETTm1 and ETTm2 provide the corresponding 15-minute series. We forecast all seven variables in the multivariate setting.

[1] https://github.com/zhouhaoyi/ETDataset

Table 6: Statistics of the nine forecasting benchmarks. The Observations column reports the number of time points in each released data file, before splitting or constructing forecasting windows.

Dataset	Variables	Frequency	Observations	Domain
ETTh1	7	1 hour	17,420	Transformer monitoring
ETTh2	7	1 hour	17,420	Transformer monitoring
ETTm1	7	15 min	69,680	Transformer monitoring
ETTm2	7	15 min	69,680	Transformer monitoring
Weather	21	10 min	52,696	Meteorology
Electricity	321	1 hour	26,304	Electricity consumption
Solar-Energy	137	10 min	52,560	Solar power generation
Exchange	8	1 day	7,588	Exchange rates
ILI	7	1 week	966	Public health

Weather. Weather contains 21 meteorological variables recorded every 10 minutes. These include atmospheric pressure, temperature, humidity, wind speed and direction, precipitation, and radiation measurements.

Electricity. Electricity records the hourly electricity consumption of 321 customers. Each channel corresponds to one customer's consumption series, and we forecast all 321 channels in the multivariate setting.

Solar-Energy. Solar-Energy records power generation from 137 photovoltaic plants in Alabama at 10-minute intervals.[2] Each plant contributes one channel. The benchmark release contains 52,560 ordered observations without absolute timestamps.

Exchange. Exchange consists of eight daily exchange-rate series, with each channel representing the exchange rate of a different currency.

ILI. The influenza-like illness (ILI) benchmark contains weekly surveillance data from the US Centers for Disease Control and Prevention (Wu et al., 2021). Its seven variables include weighted and unweighted ILI percentages, age-group case counts, total ILI counts, and the number of reporting providers.

Data splits and preprocessing. The ETT data loaders use the first 14,400 hourly observations or 57,600 observations at 15-minute resolution. These are split chronologically into training, validation, and test segments of 8,640/2,880/2,880 time points for ETTh1/ETTh2 and 34,560/11,520/11,520 for ETTm1/ETTm2. The remaining datasets use chronological 70%/10%/20% splits. Validation and test loaders prepend the required L time points from the preceding segment as historical context, while forecast targets remain within the designated split. Standardization statistics are fitted on the training segment and applied unchanged to validation and test data. Input lengths and forecasting horizons are specified in Appendix B.3 (Table 8).

B.2 Baselines and Evaluation

Baselines. We compare numerical forecasters DLinear, TimesNet (Wu et al., 2023), and PatchTST with LLM-based Time-LLM, TimeCMA, and SE-LLM (Liu et al., 2026). TimeCMA on all nine datasets and SE-LLM/Time-LLM on Solar-Energy are run locally with seed 2021. Other entries use published results matching the stated input lengths, horizons, and multivariate forecasting protocol; "–" denotes unavailable results.

[2] https://github.com/laiguokun/multivariate-time-series-data

Table 7: Model architecture and training settings for the main long-term forecasting experiments with CIRF.

Category	Parameter	Setting
Numerical forecasting backbone	Encoder layers	1
	Hidden width d_b	32
	Attention heads	2
	FFN width	32
Numerical representation branch	Encoder layers	2
	Hidden width d	128
	Attention heads	8
	FFN width	256
Prompt encoder (GPT-2)	Retained blocks	6
	Trainable blocks	6
	Hidden width	768
	Attention heads	12
	FFN width	3072
	Projection	$768 \to 128$
	Maximum prompt length	128 tokens
Training	Optimizer	AdamW
	Learning-rate schedule	Cosine annealing
	Task-layer learning rate	2×10^{-4}
	Numerical-backbone learning rate	10^{-3}
	GPT-2 learning rate	5×10^{-7}
	Weight decay	10^{-3}
	Maximum epochs	12
	Early-stopping patience	4 epochs
	Mixed precision	Enabled
	Task-layer dropout	0.1
	Forecasting loss term	Smooth L1
	Loss weights (λ_b, λ_c)	$(0.2, 0.2)$
	Loss weights (λ_r, β)	$(0.05, 0.005)$
	Gate-loss weight λ_g	0.02
	Training gate threshold τ_{train}	0.57
	Seed	2021

After checkpoint selection, the deployment threshold is selected by validation MSE over $\{0, 0.01, \ldots, 1\}$ and fixed during testing.

Evaluation metrics. We report mean squared error (MSE↓) and mean absolute error (MAE↓). For N evaluated windows, C variables, and prediction horizon H, let $e_{i,c,h} = Y_{i,c,h} - \hat{Y}_{i,c,h}$; then

$$\text{MSE} = \frac{1}{NCH} \sum_{i=1}^{N} \sum_{c=1}^{C} \sum_{h=1}^{H} e_{i,c,h}^2, \qquad \text{MAE} = \frac{1}{NCH} \sum_{i=1}^{N} \sum_{c=1}^{C} \sum_{h=1}^{H} |e_{i,c,h}|.$$

Diagnostic and robustness results first average seeds within each dataset–horizon task. Table 3 instead reports the selected seed 2021 on ETTm2 at horizon 96.

B.3 Experiment Configurations

Table 7 reports the shared model and training settings, and Table 8 gives dataset-specific input, patching, and batch settings.

Model configuration. The numerical forecasting backbone and numerical representation branch have independent parameters and different widths: the former produces the default forecast, while the latter supplies features for conditional correction and routing. The first six GPT-2 blocks are trainable, and their outputs are projected to the representation width $d = 128$.

Table 8: Dataset-specific input, patching, and batch settings for CIRF in long-term forecasting.

Dataset	Input	Patch	Stride	Micro-batch	Accum. steps	Effective batch
ETTh1	96	16	8	32	2	64
ETTh2	96	16	8	32	2	64
ETTm1	96	16	8	32	2	64
ETTm2	96	16	8	32	2	64
Weather	96	16	8	64	1	64
Electricity	96	16	8	4	8	32
Solar-Energy	96	16	8	8	4	32
Exchange	96	16	8	32	2	64
ILI	36	6	3	16	2	32

Forecasting horizons are $\{96, 192, 336, 720\}$, except for ILI, which uses $\{24, 36, 48, 60\}$. Micro-batch size is the total across active devices; effective batch size equals micro-batch size times gradient accumulation steps.

Routing and controlled-study configurations. The fixed-checkpoint routing experiments use the benchmark backbone, learning rates, weight decay, patience, and training gate threshold in Table 7. For the benchmark and routing runs, standard CIRF uses $\lambda_a = 0$ and utility margin $\delta = 0$. The ablation and CIRF-R robustness experiments instead use a numerical forecasting backbone with two layers, hidden width 128, eight attention heads, and feed-forward width 256; their patch length/stride are 16/8, weight decay is 10^{-5}, early-stopping patience is three epochs, and training gate threshold is 0.50. Sensitivity-aware training enables $\lambda_a = 0.1$ with perturbation scale 0.1. ETT attribution controls are described with Table 5, and the independently retrained Weather controls are specified in Appendix D.

Controlled studies and hardware. Controlled experiments use seeds 2021, 2022, and 2023 on two NVIDIA RTX 4090 D GPUs; the efficiency experiment uses one. The independently retrained Weather controls and their timing schedule are specified in Appendix D.

B.4 Common-Source Perturbations

We perturb normalized historical windows while leaving future labels unchanged and regenerate prompts from the perturbed histories. Gaussian noise adds zero-mean perturbations with scale s; impulse noise adds ± 3 normalized units to each point with probability s; and segment bias adds an offset of $2s$ to a contiguous segment of length $\mathrm{round}(L \min(s, 0.5))$. Each perturbation is evaluated separately at $s \in \{0.05, 0.10, 0.20\}$. CIRF-R is trained with the corresponding perturbation at $s = 0.10$, and Table 4 reports the high-intensity results at $s = 0.20$.

C Additional Forecasting Results

C.1 Complete Forecasting Results

Table 9 expands the dataset averages in Table 1 into results at each prediction horizon. CIRF achieves the lowest MSE and MAE at all four horizons on ETTh2 and Solar-Energy. On Weather and Electricity, it retains the lowest MAE across horizons, although other methods attain lower MSE at $H = 720$. ETTh1 and Exchange also show weaker long-horizon results than short-horizon results relative to competing methods. The horizon-level comparisons therefore reveal where the favorable dataset averages persist and where the ranking changes. Baseline sources follow Appendix B.

D Correction-Path Source and Cost Analysis on Weather

We extend the attribution controls to Weather to examine whether the correction path benefits from pretrained GPT-2 blocks, prompts matched to the current history, or a simpler encoding of the same statistics. These independently retrained comparisons assess the representation used for correction and its computational cost.

Table 9: Complete long-term forecasting results on nine benchmarks. MSE and MAE are reported (lower is better); Avg. gives the mean over four horizons. Bold and underlined values indicate the best and second-best results, respectively.

Dataset	H	DLinear		TimesNet		PatchTST		Time-LLM		TimeCMA		SE-LLM		CIRF	
		MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓
ETTh1	96	0.386	0.400	0.384	0.402	0.414	0.419	0.398	0.410	0.404	0.423	0.377	0.399	**0.375**	**0.393**
	192	0.437	0.432	0.436	0.429	0.460	0.445	0.451	0.440	0.455	0.449	**0.428**	0.429	0.429	**0.420**
	336	0.481	0.459	0.491	0.469	0.501	0.466	0.473	**0.451**	0.494	0.467	**0.468**	**0.451**	0.493	0.457
	720	0.519	0.516	0.521	0.500	0.500	0.488	**0.469**	0.470	0.511	0.495	0.484	**0.469**	0.538	0.497
	Avg.	0.456	0.452	0.458	0.450	0.469	0.455	0.448	0.443	0.466	0.459	**0.439**	**0.437**	0.459	0.442
ETTh2	96	0.333	0.387	0.340	0.374	0.302	0.348	0.295	0.345	0.329	0.366	0.301	0.348	**0.288**	**0.337**
	192	0.477	0.476	0.402	0.414	0.388	0.400	0.386	0.399	0.403	0.413	0.384	0.400	**0.371**	**0.386**
	336	0.594	0.541	0.452	0.452	0.426	0.433	0.419	0.429	0.437	0.442	0.437	0.440	**0.402**	**0.418**
	720	0.831	0.657	0.462	0.468	0.431	0.446	0.425	0.442	0.447	0.458	0.456	0.461	**0.412**	**0.434**
	Avg.	0.559	0.515	0.414	0.427	0.387	0.407	0.381	0.404	0.404	0.420	0.395	0.412	**0.368**	**0.394**
ETTm1	96	0.345	0.372	0.338	0.375	0.329	0.367	0.359	0.381	0.341	0.379	0.335	0.370	**0.319**	**0.351**
	192	0.380	0.389	0.374	0.387	**0.367**	**0.385**	0.383	0.393	0.385	0.403	0.393	0.406	0.379	**0.385**
	336	0.413	0.413	0.410	0.411	**0.399**	0.410	0.416	0.414	0.416	0.423	0.450	0.439	0.406	**0.404**
	720	0.474	0.453	0.478	0.450	**0.454**	**0.439**	0.483	0.449	0.485	0.459	0.555	0.488	0.475	0.444
	Avg.	0.403	0.407	0.400	0.406	**0.387**	0.400	0.410	0.409	0.407	0.416	0.433	0.426	0.395	**0.396**
ETTm2	96	0.193	0.292	0.187	0.267	**0.175**	**0.259**	0.193	0.280	0.186	0.270	0.178	**0.259**	**0.175**	0.261
	192	0.284	0.362	0.249	0.309	**0.241**	0.302	0.257	0.318	0.255	0.315	0.243	0.302	**0.241**	**0.300**
	336	0.369	0.427	0.321	0.351	0.305	0.343	0.317	0.353	0.317	0.352	0.307	0.343	**0.302**	**0.342**
	720	0.554	0.522	0.408	0.403	**0.402**	**0.400**	0.419	0.411	0.416	0.406	0.414	0.406	0.406	0.403
	Avg.	0.350	0.401	0.291	0.333	**0.281**	**0.326**	0.296	0.341	0.293	0.336	0.285	0.328	**0.281**	0.327
Weather	96	0.196	0.255	0.172	0.220	0.177	0.218	0.198	0.235	0.178	0.224	0.178	0.218	**0.169**	**0.207**
	192	0.237	0.296	**0.219**	0.261	0.225	0.259	0.240	0.269	0.225	0.264	0.226	0.258	**0.219**	**0.252**
	336	0.283	0.335	0.280	0.306	0.278	0.297	0.295	0.308	0.280	0.302	0.281	0.298	**0.277**	**0.295**
	720	**0.345**	0.381	0.365	0.359	0.354	0.348	0.368	0.353	0.355	0.351	0.361	0.348	0.355	**0.345**
	Avg.	0.265	0.317	0.259	0.286	0.259	0.280	0.275	0.291	0.259	0.285	0.261	0.280	**0.255**	**0.275**
Electricity	96	0.197	0.282	0.168	0.272	0.181	0.270	0.172	0.265	0.194	0.299	0.169	0.256	**0.165**	**0.253**
	192	0.196	0.285	0.184	0.289	0.188	0.274	0.182	0.279	0.227	0.331	0.192	0.278	**0.175**	**0.262**
	336	0.209	0.301	0.198	0.300	0.204	0.293	0.195	0.288	0.256	0.354	0.222	0.309	**0.191**	**0.278**
	720	0.245	0.333	**0.220**	0.320	0.246	0.324	0.233	0.320	0.320	0.401	0.299	0.376	0.229	**0.309**
	Avg.	0.212	0.300	0.193	0.295	0.205	0.290	0.196	0.288	0.249	0.346	0.221	0.305	**0.190**	**0.275**
Solar-Energy	96	0.290	0.378	0.250	0.292	0.234	0.286	0.405	0.444	0.305	0.363	0.213	0.252	**0.198**	**0.242**
	192	0.320	0.398	0.296	0.318	0.267	0.310	0.410	0.413	0.339	0.376	0.268	0.294	**0.233**	**0.268**
	336	0.353	0.415	0.319	0.330	0.290	0.315	0.395	0.388	0.365	0.394	0.305	0.324	**0.253**	**0.284**
	720	0.356	0.413	0.338	0.337	0.289	0.317	0.410	0.381	0.380	0.402	0.342	0.361	**0.253**	**0.281**
	Avg.	0.330	0.401	0.301	0.319	0.270	0.307	0.405	0.407	0.347	0.384	0.282	0.308	**0.234**	**0.269**
Exchange	96	0.088	0.218	0.107	0.234	0.088	0.205	0.087	0.208	0.104	0.226	0.084	0.201	**0.083**	**0.200**
	192	**0.176**	0.315	0.226	0.344	**0.176**	0.299	0.178	0.302	0.194	0.316	0.177	0.298	**0.176**	**0.296**
	336	0.313	0.427	0.367	0.448	**0.301**	**0.397**	0.338	0.422	0.363	0.442	0.323	0.412	0.328	0.414
	720	0.839	0.695	0.964	0.746	0.901	0.714	**0.819**	**0.681**	0.956	0.747	0.855	0.699	0.849	0.688
	Avg.	**0.354**	0.414	0.416	0.443	0.366	0.404	0.356	0.403	0.404	0.433	0.360	0.402	0.359	**0.399**
ILI	24	2.398	1.040	2.317	0.934	2.335	0.989	2.383	1.004	4.626	1.579	2.979	1.035	**2.202**	**0.897**
	36	2.646	1.088	**1.972**	0.920	2.561	1.035	2.390	0.993	4.438	1.532	2.344	0.970	2.220	**0.916**
	48	2.614	1.086	**2.238**	**0.913**	2.465	1.022	2.394	1.003	4.247	1.485	2.397	0.961	2.286	0.934
	60	2.804	1.146	**2.027**	0.928	2.189	0.997	2.562	1.049	4.276	1.488	2.148	0.941	2.131	**0.920**
	Avg.	2.615	1.090	**2.139**	0.924	2.388	1.011	2.432	1.012	4.397	1.521	2.467	0.977	2.210	**0.917**

Setup. We use all 21 variables, input length 96, horizons $H \in \{96, 336\}$, and a chronological 70/10/20 split with standardization fitted on training data. Five configurations are independently trained on the full training split with seeds 2021, 2022, and 2023, yielding 30 runs; shared modules have matched initial weights. The numerical forecasting backbone has one layer, hidden width 32, two attention heads, and feed-forward width 32; weight decay is 10^{-3}, early-stopping patience is four epochs, and the training gate threshold is 0.57. AdamW retains the clean Weather learning rates of $2 \times 10^{-4}/10^{-3}/5 \times 10^{-7}$ for task layers, numerical backbone, and GPT-2 blocks. All runs use micro-batches of eight with eight-step gradient accumulation and at most 12 epochs. Checkpoints minimize validation MSE at routing threshold 0.57 for gated variants; the threshold is then selected from $\{0, 0.01, \ldots, 1\}$ on validation data and fixed for testing.

Controls. *Numeric-only* independently trains the same numerical forecaster and normalization layers with a Huber objective. *Randomized GPT-2 blocks* reinitializes the six trainable Transformer blocks while retaining the pretrained, frozen token/position embeddings and final layer normalization. *Unmatched prompts* replaces dynamic prompt statistics with those from a non-overlapping history of the same variable during training, validation, and testing; static descriptions remain unchanged. Donor histories end no later than the current input's start, with mappings fixed per seed.[3] *Stats-MLP* replaces tokenization and GPT-2 with a $4 \to 256 \to 768$ MLP with a GELU activation. Its inputs are the minimum, maximum, median, and ± 1 trend code from the same normalized history, with continuous statistics rounded to six decimals as in the prompt template. It retains the prompt projection, conditional predictor, residual head, gate, and clean-training losses.

Table 10: Independently retrained controls on Weather with input length 96 and the full training split. Errors are mean $\pm$ sample standard deviation over seeds 2021, 2022, and 2023. Values use three decimals; SD 0.000 means below 5×10^{-4}. Bold marks the lowest unrounded mean in each column.

Configuration	$H = 96$		$H = 336$	
	MSE↓	MAE↓	MSE↓	MAE↓
Full CIRF	0.171 ± 0.002	0.209 ± 0.002	0.277 ± 0.002	0.296 ± 0.001
Numeric-only	0.173 ± 0.001	0.210 ± 0.000	0.276 ± 0.000	$\mathbf{0.294} \pm 0.001$
Randomized GPT-2 blocks	0.172 ± 0.000	0.209 ± 0.001	0.277 ± 0.002	0.295 ± 0.001
Unmatched prompts	0.172 ± 0.001	0.209 ± 0.001	0.277 ± 0.002	0.295 ± 0.002
Stats-MLP	$\mathbf{0.170} \pm 0.000$	$\mathbf{0.208} \pm 0.002$	$\mathbf{0.275} \pm 0.002$	0.295 ± 0.004

Forecasting accuracy. Table 10 shows that Full CIRF improves on Numeric-only in both mean MSE and MAE at $H = 96$, but has higher errors at $H = 336$. Randomized blocks and unmatched prompts produce small changes whose direction also reverses across horizons. Stats-MLP has the lowest mean MSE at both horizons and the lowest MAE at $H = 96$, whereas Numeric-only has the lowest MAE at $H = 336$. These results show no consistent forecasting advantage from pretrained GPT-2 blocks or matched dynamic prompt statistics under this protocol. They separate the usefulness of a correction mechanism from the choice of representation that supplies it: a compact encoding of the same statistics can be competitive without GPT-2.

Cost measurement. Each run uses one RTX 4090 D, with up to four concurrent jobs sharing the host. We time ten synchronized float32 forward passes on the first test batch of eight after three warm-up passes and retain the median. Timing includes online statistics, tokenization, prompt encoding, and forecasting where applicable, excluding data loading and host-to-device transfer. Training time includes validation and checkpoint saving; early stopping allows training lengths to differ.

Accuracy and cost. Stats-MLP uses fewer than one million parameters and takes about 2 ms per batch, compared with 114–130 ms for Full CIRF in Table 11. Together with its competitive errors, this makes the representation used for correction a material design choice. In this common-source setting, the benefit of the derived view should be judged by its forecasting contribution over the

[3] For early training windows without sufficient past history, the donor is drawn from the training split after the current input and target. Validation and test donors always precede the current input.

Table 11: Model size and observed runtime for the Weather controls. M denotes 10^6 parameters. Inference uses float32 and batches of eight on one RTX 4090 D; reported values aggregate per-run median batch times as mean $\pm$ sample standard deviation across three seeds. Training and validation times use the same across-seed summary and include checkpoint saving. Measurements follow the shared-host schedule described in the text.

Configuration	H	Parameters (M)		Inference (ms/batch)	Train + val. (min)
		Total	Trainable		
Full CIRF	96	82.543	43.158	114.467 ± 1.864	128.918 ± 31.554
	336	82.666	43.281	130.136 ± 21.622	142.039 ± 9.946
Numeric-only	96	0.044	0.044	0.738 ± 0.021	10.809 ± 1.023
	336	0.137	0.137	0.675 ± 0.070	9.825 ± 2.183
Randomized GPT-2 blocks	96	82.543	43.158	114.269 ± 1.455	141.123 ± 10.490
	336	82.666	43.281	115.561 ± 0.620	120.888 ± 6.592
Unmatched prompts	96	82.543	43.158	114.371 ± 0.765	134.314 ± 32.356
	336	82.666	43.281	115.421 ± 0.395	145.675 ± 28.451
Stats-MLP	96	0.829	0.829	2.014 ± 0.059	23.668 ± 7.768
	336	0.953	0.953	1.984 ± 0.007	22.910 ± 3.927

numerical default and its measured cost; the use of language pretraining alone does not establish an additional benefit. The runtime comparison pertains to the stated shared-host schedule.