Robust Residual Correction via Selective Deployment for Time Series Forecasting

Jianxiang Xie[1] **Yuncheng Hua**[1,4] **Mingyue Cheng**[2] **Flora D. Salim**[1,4] **Hao Xue**[3,4*]
[1]School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
[2]State Key Laboratory of Cognitive Intelligence, School of Computer Science and Technology, University of Science and Technology of China, Hefei, China
[3]The Hong Kong University of Science and Technology (Guangzhou)
[4]DataCruiser

Abstract

Post-hoc residual correction can reduce forecasting error, but always-on updates often fail to generalize: when residual signals are weak or noisy, expressive correctors can overfit and even increase the overall forecasting errors. We introduce **CRC**, a robust residual correction framework that couples a hybrid corrector with a validation-calibrated *selective deployment* rule. CRC proposes corrections via a conservative ridge "floor" plus conditional nonlinear refinement, but deploys them only when held-out evidence indicates net error reduction over a frozen baseline forecaster. CRC treats *when* to revise a forecast as a validation-calibrated deployment decision, not only how to fit residuals. Across standard long-sequence benchmarks and multiple backbones, CRC improves horizon-averaged MSE and MAE in most dataset–backbone pairs, with particularly large gains on complex datasets (e.g., over 20% relative MSE reduction on Traffic). We additionally report the non-degradation rate (NDR) as a stability diagnostic of harmful updates and ablate the validation-calibrated "firewall" that stabilizes MSE gains. *Code:* `https://github.com/Xavian1-oss/CRC.git`.

1 Introduction

Multivariate time series forecasting underpins decisions in transportation, energy, finance, and urban systems. Modern pipelines produce vector-valued trajectories across many horizons. Accuracy is judged not only globally but channel-by-channel and step-by-step when planners route traffic, schedule power, or monitor anomalies. Deep learning has improved average accuracy [11, 27, 5, 24], yet prediction errors typically vary across channels and horizons. Where forecasts gate operations or downstream risk assessment, localized failures matter even when headline metrics improve.

Because retraining large forecasters is costly, *post-hoc residual correction* is an appealing add-on: one fits an auxiliary module on baseline errors while keeping the backbone frozen [4, 28]. The usual recipe regresses residual evidence against targets and applies the resulting update *by default* at test time. That *always-on* regime is fragile: under weak residual structure relative to noise or misspecification, expressive maps can chase idiosyncratic fluctuations and *increase* test mean squared error (MSE) relative to the baseline—even when training loss looks favorable. This tension is the *Corrector's Dilemma*: bias that could be reduced in aggregate must still be shielded from noise amplification at deployment time, so the operative question includes *whether* and *when* to apply an update, not only how to parameterize one. Importantly, the difficulty is not resolved by fitting residuals more aggressively on the training split; without a deployment gate, stronger correctors can amplify the mismatch between in-sample fit and held-out error.

*Corresponding author: Hao Xue (haoxue@hkust-gz.edu.cn).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

We study selective deployment under deterministic point forecasting from a frozen backbone—the evaluation protocol adopted by standard multivariate forecasting benchmarks [34, 30]—with squared error summarized at horizon and variable granularity. **CRC** consumes the historical window and baseline prediction, builds direction-aware residual features via a ridge "floor" and conditional nonlinear refinement, and passes proposals through a deployment rule calibrated on validation and frozen at test time; it selects among baseline and refinement paths using held-out evidence (*without* oracle access to test residuals). Our goal is improved test accuracy under the benchmark protocol and honest diagnostics when updates fail locally instead of distribution-shift guarantees beyond what validation calibration reasonably supports (limitations are discussed in Section 5).

Contributions.

- **Perspective.** Residual correction is framed as deciding *deployment* rather than maximizing in-sample residual fit alone: expressive updates carry generalization risk when residual evidence is thin, motivating a validation-calibrated veto over always-on refinement (complementary to coverage-based or abstention-focused methods; see Section 2).
- **Method.** CRC couples a hybrid corrector with a granular selector: direction-aware features, ridge-backed linear recovery, guarded nonlinear refinement, and per variable–horizon deployment choices calibrated on validation and frozen for testing. The deployment rule composes simple, interpretable filters (direction consistency, magnitude clipping, local path selection, and shrink-to-base margins) so that refinements are released only where validation supports net benefit.
- **Empirics and diagnostics.** Across seven benchmarks and four backbones, CRC improves horizon-averaged MSE and MAE in most dataset–backbone combinations, with larger margins where cross-channel residual structure appears informative. We report non-degradation rate (NDR) alongside error metrics to quantify how often corrections worsen local predictions, and use ablations to isolate the deployment "firewall" relative to architectural choices.

Granularity. Operations often care both about pooled error and about individual variable–horizon behavior. Aggregate MSE may improve while silently degrading forecasts on a subset of sensors or future steps (precisely where practitioners notice instability or incur uneven costs). Our non-degradation rate (NDR) estimates how frequently a correction leaves each (i, h) unit no worse than the baseline (ties allowed), exposing harmful updates that averages can obscure. CRC evaluates and gates deployment at the same (i, h) units tracked by these diagnostics, rather than trusting a single pooled residual regression to authorize updates everywhere.

2 Related Work

Most post-hoc pipelines emphasize *how* to model a residual map $\Delta(\cdot)$ and then deploy $\hat{Y}^{\mathrm{base}} + \Delta$ by default (*always-on* correction), which can harm units that are already accurate when residuals are noisy. Residual modules and instance-aware revisions [4, 12] largely pursue average-error reduction on frozen forecasters. CRC adds a deploy-time layer $\mathcal{I}(\cdot)$ *calibrated on validation*: it chooses whether to deploy a refinement at fine granularity, complementary to stacking another regressor wholesale (Section 3), and plugs into standard long-horizon backbones (e.g., DLinear, PatchTST, TimesNet [33, 17, 29]).

Instance-aware post-hoc revision (PIR [12]) targets difficult instances for retrieval-driven repair. CRC instead stabilizes pointwise error by *gating deployment* so harmful updates are suppressed rather than universally applied. Test-time adaptation and forecaster-selection lines such as TAFAS [8], PETSA [14], FAC [25], SOLID [2], and FFORMA [16] typically update or choose models under test-time feedback. CRC keeps the backbone frozen, uses no test labels, and only decides whether to release an additive residual correction; NDR is reported as a local non-degradation check. δ-Adapter [10] and related adapters illustrate safety-motivated post-hoc adaptation. CRC focuses narrowly on deploying (or withholding) additive residual corrections tied to frozen $\hat{Y}^{\mathrm{base}}$.

Reliability-focused learning spans conformal prediction [22], selective prediction [3], and robust objectives [15]; these emphasize coverage/sets or worst-case risk rather than selectively releasing point refinements after $\hat{Y}^{\mathrm{base}}$ is fixed. Sequential conformal and calibration methods [23, 31, 32] likewise differ in target from *when-to-deploy* corrections aimed at stabilizing deterministic MSE. Neural

basis-expansion architectures such as N-BEATS [18] optimize accuracy but rarely attach an explicit veto rule for harmful residual updates at deployment time. Structure-aware representations [1, 21, 20] inform inductive bias; we use directional pairwise evidence without claiming causal discovery.

3 Method

Throughout this section, *degradation* refers specifically to correction-induced performance deterioration relative to the baseline forecast. The exposition follows Fig. 1: oracle and features (Sections 3.1 to 3.3), then deployment (Section 3.4).

We formulate residual correction as a *selective deployment problem*: CRC seeks lower held-out MSE by deploying updates only where validation indicates net benefit, a calibration *design goal*, not an unconditional guarantee at test time (Appendix A). The centerpiece is thus both a correction map Δ and a frozen deployment rule $\mathcal{I}(\cdot)$ that decides *whether* each proposed update is applied. Given a historical sequence $W \in \mathbb{R}^{B\times P\times N}$, where B is the batch size, P is the look-back window, and N is the number of variables, together with a baseline forecast $\hat{Y}^{\text{base}} \in \mathbb{R}^{B\times H\times N}$ for horizon H, the goal is not to apply correction indiscriminately, but to deploy it *only when correction is expected to be beneficial*. Accordingly, CRC is designed to answer two coupled questions: (i) *what correction should be proposed*, and (ii) *when should that correction be applied*?

We formalize the corrected forecast as

$$\hat{Y} = \hat{Y}^{\text{base}} + \mathcal{I}(\Delta), \tag{1}$$

where Δ is a candidate residual correction and $\mathcal{I}(\cdot)$ denotes a *single* validation-calibrated deployment mapping, frozen on validation and applied identically at test time. At the same variable–horizon granularity used by our stability diagnostic (NDR, Section 4), $\mathcal{I}$ decides how much of each proposed update is released instead of concatenating unrelated knobs. Operationally, $\mathcal{I}$ is an empirical surrogate for a *risk-aware selector*: deploy a proposed Δ when validation evidence supports that the expected MSE benefit outweighs the risk of noise amplification under the firewall constraints. Our objective is therefore not aggressive residual fitting, but robust refinement: deploy corrections only when validation evidence indicates a favorable improvement–risk tradeoff, so as to reduce test MSE while limiting harmful updates.

Ideal objective (for intuition). Here $\text{Error}(\cdot)$ denotes the per-unit loss used to judge whether a correction is beneficial (e.g., squared error for MSE or absolute error for MAE) at the deployment granularity.

$$\text{Error}(\hat{Y}) \le \text{Error}(\hat{Y}^{\text{base}}) \tag{2}$$

whenever the deployment rule approves the correction. This inequality is an *idealized* design principle (not a guarantee). CRC approximates it by calibrating and freezing the deployment rule on validation.

Overview. CRC follows a robust-by-design pipeline (Fig. 1) with three components: (1) a direction-aware residual representation that exposes recoverable structure, (2) a conservative–conditional corrector that proposes candidate updates, and (3) a validation-calibrated deployment rule that determines whether and how those updates are deployed.

3.1 Residual Correction as Selective Deployment

Residuals contain both *recoverable structure* and *irreducible noise*, expressive correctors may harm accuracies by chasing spurious fluctuations, that is, the **Corrector's Dilemma** of Section 1.

CRC adopts a selective deployment view. Instead of always applying a learned correction, we first generate a *candidate update* and then apply it only under a validation-calibrated deployment rule. Formally, let Δ_i denote the candidate correction for variable i. An ideal corrector would follow the oracle rule

$$\hat{Y}_i = \begin{cases} \hat{Y}_i^{\text{base}} + \Delta_i, & \text{if } \text{Error}(\hat{Y}_i^{\text{base}} + \Delta_i) \le \text{Error}(\hat{Y}_i^{\text{base}}), \\ \hat{Y}_i^{\text{base}}, & \text{otherwise.} \end{cases} \tag{3}$$

Since this oracle is unavailable at deployment, CRC approximates it through a validation-calibrated deployment rule. This perspective turns residual correction from an unconstrained regression problem into a *decision-aware refinement problem*.

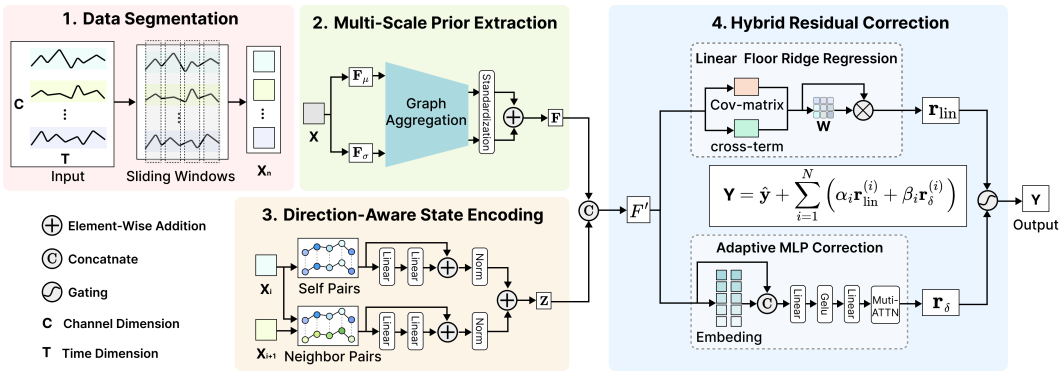

Figure 1: Overview of the CRC framework. CRC first extracts structured residual representations, then proposes conservative and conditional correction components, and finally applies a validation-calibrated deployment rule that deploys corrections only when held-out evidence supports them.

3.2 Structured Residual Representation

A selective deployment rule is only meaningful if the candidate correction is grounded in structured residual evidence. Directly modeling the raw residual $R = Y - \hat{Y}^{\text{base}}$ is unreliable, because it entangles correctable bias with random noise. CRC therefore first constructs a direction-aware representation designed to expose residual structure that is stable enough to support conservative correction. The encoder feeds both the hybrid corrector and the deployment rule that may veto harmful updates.

Given prior adjacency $A \in \{0,1\}^{N\times N}$ ($A_{ij}=1$ if j may influence i), a shared module f_{pair} processes self pairs and cross pairs. We write U_i for the learned pairwise interaction embedding (distinct from the raw residual R above):

$$(U_i, \alpha_{i\to i}) = f_{\text{pair}}(W_{:,:,i}, W_{:,:,i}), \tag{4}$$

$$(U_j, \alpha_{j\to i}) = f_{\text{pair}}(W_{:,:,i}, W_{:,:,j}), \quad A_{ij}=1. \tag{5}$$

Gates α are direction-aware (via $\tanh$) but are not intended for causal identification. We aggregate

$$Z_i = \frac{1}{|\mathcal{N}(i)| + \alpha_{i\to i,\text{const}}} \left(\alpha_{i\to i} U_i + \sum_{j:A_{ij}=1} \alpha_{j\to i} U_j \right), \tag{6}$$

where $\alpha_{i\to i,\text{const}} > 0$ is a fixed stabilizing constant (set to 1 in our implementation) that prevents division by zero when $\mathcal{N}(i)$ is empty and keeps the normalization well-defined. We then concatenate multi-scale statistical priors F (means, trends, correlations, band energy; neighborhood pooled with A_{norm}; see Appendix B.1) to obtain

$$Z_{\text{aug}} = [Z, F]. \tag{7}$$

Constructing the adjacency used by f_{pair}. The binary adjacency A specifies which ordered pairs $(j \to i)$ are eligible for cross-node messaging. In our implementation, A is built once from training data by correlation-based K-nearest neighbors over variables (K=5; see Section B.2), then fixed for validation and testing. This yields a sparse prior over interactions without assuming a known causal graph; the learnable gates α modulate which edges are active for a given window.

3.3 Conservative–Conditional Correction

Given the structured features Z_{aug}, CRC constructs a candidate correction by decomposing it into a *conservative component* and a *conditional refinement component*. This decomposition reflects a key design principle: stable, linearly explainable residual structure should be recovered by default, while more expressive nonlinear correction should only be trusted conditionally.

For each variable i, CRC defines the candidate correction as

$$\Delta_i = \Delta_i^{\text{base}} + \Delta_i^{\text{ref}}, \tag{8}$$

where Δ_i^{base} is a conservative baseline recovery term and Δ_i^{ref} is a higher-order refinement term.

Conservative baseline recovery. CRC uses ridge regression [7] to recover stable, linearly explainable bias:

$$\boldsymbol{\beta}_i = \arg\min_{\boldsymbol{\beta}} \left(\|\mathbf{r}_i - Z_{\text{aug},i}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \right), \tag{9}$$

where $\mathbf{r}_i = \mathbf{Y}_i - \hat{\mathbf{Y}}_i^{\text{base}}$ is the residual of the baseline forecast. The resulting conservative correction is

$$\Delta_i^{\text{base}} = Z_{\text{aug},i}\boldsymbol{\beta}_i. \tag{10}$$

This term acts as a stable recovery floor, capturing residual structure that is both interpretable and robust.

Conditional refinement. To model higher-order residual patterns beyond the linear floor, CRC employs a lightweight MLP:

$$\Delta_i^{\text{ref}} = f_\theta(Z_{\text{aug},i}). \tag{11}$$

This refinement is expressive but potentially risky, since nonlinear updates are more vulnerable to overfitting noise. Accordingly, Δ_i^{ref} is not deployed directly. Instead, it is passed to the deployment rule described next.

3.4 Validation-Calibrated Deployment Rule

The role of $\mathcal{I}(\cdot)$ is to approximate the ideal oracle in Eq. (3). We treat $\mathcal{I}$ as *one* test-frozen deployment policy: it composes (i) *stabilizing transforms* of the nonlinear proposal (direction checks and magnitude clipping) with (ii) a *baseline-inclusive* discrete path choice at each (i, h) unit, and (iii) a validation-level *margin* that defaults to the baseline when estimated improvements are too small to be reliable. This factorization targets complementary failure modes (sign flips, heavy-tailed updates, cross-unit heterogeneity, and selection noise) while keeping a single object (i.e., the released update) well defined at the same granularity as NDR.

3.4.1 Factorization of the deployment rule (robust deployment)

The following items are the operational steps that instantiate the same $\mathcal{I}$ (each detailed at training/validation time). Clipping thresholds and deployment weights are the primary algebraic objects:

- **Direction consistency (stabilize):** accept nonlinear refinement only when aligned with the post-linear residual direction (suppresses harmful sign flips).
- **Magnitude control (stabilize):** clip candidates with validation-calibrated thresholds τ:

  $$\Delta^{\text{clip}} = \text{clip}(\Delta, -\tau, \tau). \tag{12}$$

  Here Δ is the gated candidate entering clipping. Appendix A uses an equivalent linear-floor plus clipped-nonlinear form for analysis.
- **Local path selection (discrete, per (i, h)):** freeze whichever of linear-only vs. hybrid correction wins on validation at each evaluation unit.
- **Global validation gating (margin):** deploy only if validation gains exceed a margin; otherwise revert to the baseline channel.

The deployed forecast combines baseline and (possibly zeroed) corrections:

$$\hat{Y} = \hat{Y}^{\text{base}} + w_1\Delta^{\text{base}} + w_2\Delta^{\text{clip}}, \tag{13}$$

where (w_1, w_2) are chosen on validation per unit (including zeros when a path is disabled), consistent with selecting among candidates that include the baseline (Proposition A.3). Thresholds τ, local choices, and weights are fit on validation and *frozen* for test (see Appendix B.2).

3.4.2 Interpretation as Oracle Approximation

The above constraints should not be viewed as isolated heuristics. Instead, they compose the single mapping $\mathcal{I}$ in Eq. (1) and provide *design-motivated* approximations of the unavailable oracle rule in Eq. (3) along four axes (direction, magnitude, locality, and validation reliability). We do not claim these steps are the exact solution of a single constrained optimization problem; rather, they instantiate selective deployment by coupling a candidate update with a frozen, validation-calibrated decision. Appendix A provides supporting statements under explicit assumptions.

3.5 Complexity and Deployment

Training cost is dominated by the encoder over length-P windows and pairwise interaction terms; in the worst case this scales as $O(N^2)$, while our implementation uses a sparse K-NN adjacency ($K{=}5$) to restrict eligible pairs. Ridge and the MLP are lightweight. At inference, CRC adds only a small post-hoc module on top of a frozen backbone forecast: a single pass through the encoder/MLP and a frozen per-unit deployment lookup (no iterative optimization at test time). A detailed complexity discussion is deferred to Appendix B.4.

4 Experiments

4.1 Experimental Setup

Datasets and Backbones. We evaluate CRC on seven standard long-sequence benchmarks [34, 30] at horizons $\{96, 192, 336, 720\}$. **ETTh1/ETTh2** and **ETTm1/ETTm2** include electricity load and oil temperature at hourly and 15-minute resolutions, **Weather** has 21 meteorological channels, **Electricity** and **Traffic** are large collections of power and road-occupancy series. Together they span relatively smooth dynamics to highly non-stationary behavior.

CRC is applied as a plug-in post-hoc corrector to four forecasting backbones: TimeXer, TimesNet, PatchTST, and DLinear [26, 29, 17, 33]. Unless otherwise noted, we report horizon-averaged MSE (our main metric) and MAE averaged over three random seeds. We additionally report NDR as a diagnostic signal of harmful updates and use it to interpret when selective deployment stabilizes test-time MSE gains.

Benchmark scope. Our primary head-to-head comparison is against PIR [12] under the standard long-horizon splits with identical backbone checkpoints for both methods (Section 4.3). Adapter- and calibration-style systems often use different objectives or protocols; we relate them qualitatively in Section 2.

Evaluation Protocol. Our primary goal is to improve forecasting accuracy with **MSE as the main metric**. We report horizon-averaged MSE (with MAE as a secondary metric) and additionally use **Non-Degradation Rate (NDR)** as a stability diagnostic: the fraction of $(\text{node}, \text{horizon})$ pairs for which the corrected prediction is no worse than the baseline (ties count as non-degrading). Unless stated otherwise, we report test-set NDR.

Training protocol. Implementation details are in Appendix B.2: two-stage training (encoder pre-training, then hybrid corrector), AdamW/Adam, correlation-based K-NN adjacency ($K{=}5$), $Q{=}0.80$, $\varepsilon{=}0.01$. PIR and CRC use the same standard splits and frozen backbone checkpoints; post-hoc modules are tuned only on training/validation folds.

Deployment calibration. At each $(\text{variable}, \text{horizon})$ unit, CRC selects on the validation split among three frozen candidates (i.e., the backbone forecast, ridge-only correction, and hybrid correction) by lowest validation loss, then fixes that path for test-time deployment without access to test labels. This matches the validation-calibrated rule in Section 3.4 (Eq. (13)). Quantile clipping and shrink-to-base use the defaults above. Full instantiation is in Appendix B.3.

4.2 Main Results: Robust MSE Improvements via Selective Deployment

We report (i) horizon-averaged accuracy over seven benchmarks and four backbones (Table 1), (ii) a deployment ablation with test-time NDR on Electricity–DLinear at $H{=}192$ (Table 2), and (iii) a controlled G0–G5 ladder on three representative single-horizon settings that isolates always-deploy vs. selective policies (Table 3, seed 2021, not horizon-averaged). Appendix D complements (iii) with seed variability on B1. PIR comparison and calibration sensitivities follow in later subsections.

Across our benchmarks, CRC improves MSE/MAE in most settings. We first summarize headline accuracy, then deployment mechanisms.

Table 1: Horizon-averaged MSE and MAE improvements of CRC across datasets and backbones (averaged over horizons 96, 192, 336, 720). Improvements are relative to the baseline and positive values indicate gains. Full horizon-wise results are reported in Appendix Table 7.

Dataset	Model	MSE (Baseline)	MSE (CRC)	Imp MSE (%)	MAE (Baseline)	MAE (CRC)	Imp MAE (%)
ETTh1	DLinear	0.4603	0.4582	**0.46**	0.4568	0.4554	**0.29**
	PatchTST	0.4478	0.4468	**0.22**	0.4476	0.4454	**0.50**
	TimeXer	0.4441	0.439	**1.14**	0.4451	0.444	**0.24**
	TimesNet	0.4565	0.4562	**0.06**	0.4528	0.4525	**0.06**
ETTh2	DLinear	0.5641	0.5383	**4.57**	0.5194	0.497	**4.32**
	PatchTST	0.3807	0.3794	**0.35**	0.4092	0.4078	**0.36**
	TimeXer	0.3769	0.3774	-0.13	0.4006	0.4016	-0.24
	TimesNet	0.4082	0.4082	**0.01**	0.4224	0.4219	**0.10**
ETTm1	DLinear	0.404	0.3935	**2.60**	0.4077	0.405	**0.65**
	PatchTST	0.3888	0.3863	**0.64**	0.4029	0.4019	**0.24**
	TimeXer	0.3817	0.3846	-0.76	0.3968	0.3982	-0.35
	TimesNet	0.4199	0.4142	**1.37**	0.4208	0.4195	**0.33**
ETTm2	DLinear	0.3547	0.3375	**4.86**	0.4018	0.3808	**5.21**
	PatchTST	0.2923	0.2888	**1.19**	0.3353	0.333	**0.68**
	TimeXer	0.2735	0.2688	**1.74**	0.3214	0.32	**0.44**
	TimesNet	0.2947	0.2918	**1.01**	0.3328	0.3326	**0.07**
Electricity	DLinear	0.2178	0.1801	**17.30**	0.3082	0.2775	**9.95**
	PatchTST	0.2038	0.1772	**13.05**	0.2943	0.2722	**7.49**
	TimeXer	0.1724	0.1634	**5.28**	0.2707	0.2609	**3.65**
	TimesNet	0.1967	0.1864	**5.22**	0.2963	0.2877	**2.92**
Traffic	DLinear	0.6384	0.5042	**21.03**	0.3946	0.3184	**19.31**
	PatchTST	0.4914	0.4646	**5.47**	0.3151	0.2973	**5.66**
	TimeXer	0.4705	0.4653	**1.11**	0.2875	0.2804	**2.48**
	TimesNet	0.6322	0.6039	**4.47**	0.3353	0.3138	**6.40**
Weather	DLinear	0.2654	0.2385	**10.13**	0.317	0.2833	**10.61**
	PatchTST	0.2582	0.2362	**8.52**	0.2796	0.2676	**4.33**
	TimeXer	0.2406	0.2318	**3.67**	0.2707	0.2651	**2.07**
	TimesNet	0.2631	0.2401	**8.75**	0.2888	0.2746	**4.94**

4.2.1 Accuracy: MSE/MAE Improvements

Table 1 reports forecasting accuracy with CRC relative to each backbone. Across the benchmark suite, CRC improves MSE and MAE in most dataset–backbone pairs; in Table 1 (7 datasets $\times$ 4 backbones), CRC improves horizon-averaged MSE in 26 out of 28 dataset–backbone pairs (the two exceptions are TimeXer on ETTh2 and ETTm1). Table 1 also contains a minority of flat or slightly negative entries, which we interpret as weak residual evidence or backbone-specific interactions rather than systematic failure of selective gating. Validation-calibrated selective deployment allows strong gains wherever structured residuals dominate, while reducing the frequency of harmful updates (as reflected by NDR where we evaluate it, Table 2).

The gains are particularly pronounced on structurally complex benchmarks such as **Traffic**, **Electricity**, and **Weather**, where residual patterns are more likely to contain exploitable structure. In these settings, CRC benefits from combining conservative recovery with conditional refinement, producing stronger improvements while maintaining a stable correction behavior.

Table 2: Ablation of CRC components on **Electricity** (**DLinear** backbone, horizon H=192). **MSE** is reported at this horizon (lower is better). **NDR** is higher (better).

Variant	MSE ↓	NDR ↑
DLinear baseline	0.198	—
CRC (full, ours)	**0.173**	**95.0**
(1) ***Hybrid corrector***		
1a. w/o ridge floor (MLP-only)	0.181	70.0
1b. w/o MLP Δ (ridge-only)	0.192	94.0
(2) ***Safety firewall***		
2a. w/o quantile clipping	0.176	85.0
2b. w/o direction gating	0.175	88.0
2c. w/o point-wise selection	0.174	90.0
2d. w/o shrink-to-base blending	0.174	82.0
(3) ***Direction-aware encoder***		
3a. w/o graph (self-only)	0.185	93.0
3b. w/o encoder (simple features)	0.195	90.0

By contrast, when residual evidence is weak (e.g., **ETTm1**), CRC tends to abstain and yields smaller gains rather than forcing updates that can hurt MSE.

Table 3: Controlled deployment ladder (G0–G5) on B1–B3 (seed 2021). B1/B2/B3: ETTh1+DLinear at pred_len 96/336 and Weather+TimesNet at 96. **Not** horizon-averaged like Table 1. G0 baseline, G1 always-deploy (blend search off; τ/direction gate retained), G2 ridge+validation gate only (no MLP), G3 w/o direction gate, G4 deploy-only encoder, G5 full CRC. Lower MAE/MSE and higher NDR (%) are better.

Config	B1 MAE	B1 MSE	B1 NDR	B2 MAE	B2 MSE	B2 NDR	B3 MAE	B3 MSE	B3 NDR
G0 Baseline	0.4108	0.3962	—	0.4654	0.4874	—	0.2184	0.1685	—
G1 Always-on	0.4738	0.4665	8.2	0.5689	0.6328	10.6	0.2326	0.1803	33.9
G2 Ridge+val gate	0.4097	0.3941	97.9	0.4693	0.4920	97.1	0.2111	0.1626	90.0
G3 No dir gate	0.4079	0.3865	81.8	0.4684	0.4786	78.5	0.2098	0.1611	81.5
G4 Deploy-only	0.4097	0.3941	62.1	0.4592	0.4838	77.8	0.2081	0.1584	79.4
G5 Full CRC	**0.3914**	**0.3780**	**80.3**	**0.4392**	**0.4612**	**79.7**	**0.1903**	**0.1408**	**78.5**

4.2.2 Controlled Comparison: Always-on vs. Selective Deployment

To isolate the effect of deployment decisions, we compare the frozen baseline (G0), an always-deploy corrector under the controlled protocol (G1), and four gated or ablated variants (G2–G5) in Table 3. On B1 (ETTh1, DLinear, $H{=}96$), G1 increases MSE from 0.3962 to 0.4665 with NDR 8.2%, whereas the ridge-only validation gate (G2) reaches NDR 97.9% but only modest MSE improvement (0.3941). Full CRC (G5) reduces MSE to 0.3780 while maintaining NDR 80.3%. Similar trends hold on B2 and B3. These results show that conservative gating alone is insufficient for the strongest accuracy gains, while unrestricted deployment of corrections can be harmful, combining selective deployment with conditional refinement yields a more favorable accuracy–stability tradeoff.

4.2.3 Diagnostics: Harmful-Update Frequency (NDR)

Table 2 shows that the full CRC model achieves a high NDR of 95.0% on the DLinear–Electricity setting, indicating that the proposed deployment rule avoids degradation in the large majority of variable–horizon cases. Removing any component of the deployment rule reduces NDR substantially, indicating that stability arises from the overall selective deployment policy rather than a single heuristic.

4.2.4 Takeaway

The three tables highlight different aspects of CRC: overall forecasting accuracy (Table 1), the contribution of individual components (Table 2), and the cost of deploying corrections indiscriminately (Table 3). Together they indicate that gains depend on *whether* refinements are released (the deployment stack), not only on how expressive the residual head is. Appendix D reports mean $\pm$ std seed variability on the same representative B1 setting (Table 6).

4.2.5 Operational tradeoffs and calibration robustness

Appendix C studies two deployment-time sensitivities: inference-time τ rescaling, which traces an accuracy–stability tradeoff (test MSE vs. test NDR) via clipping aggressiveness, and the validation-ratio budget used to fit the frozen rule (5%–20%), where headline metrics remain relatively stable.

4.3 Comparison with PIR: Effectiveness of Validation-Calibrated Deployment

We next compare CRC with the recent post-forecasting identification and revision (PIR) framework [12], a strong post-hoc correction baseline that performs instance-level retrieval and revision, but does not explicitly calibrate correction through a validation-calibrated deployment rule designed to stabilize test-time error.

Table 4 reports relative error differences $\Delta = \text{PIR} - \text{CRC}$ (positive favors CRC). CRC is lower on 18 of 28 dataset–backbone pairs (PIR lower on 9, one tie). The two methods address different failure modes: PIR selects which instances to revise, while CRC gates whether a proposed correction is deployed, so the comparison is not meant as a universal ranking. Appendix C.1 further reports horizon-wise NDR on Weather, where CRC often yields stronger stability diagnostics when harmful-update suppression is prioritized.

Table 4: Horizon-averaged relative forecasting error differences between PIR and CRC (Δ = PIR − CRC; positive indicates CRC better). **Winner** summarizes the majority sign of Δ across backbone–horizon pairs for each dataset. Full horizon-wise results (including MAE) are reported in Appendix Table 8.

Dataset	TimeXer	TimesNet	PatchTST	DLinear	Winner
ETTh1	−0.0223	0.0000	−0.0036	−0.0217	PIR
ETTh2	−0.0076	0.0053	0.0141	−0.1673	CRC
ETTm1	0.0116	0.0116	0.0042	−0.0065	CRC
ETTm2	0.0205	0.0138	0.0102	−0.0537	CRC
Weather	0.0117	0.0192	0.0371	0.0267	CRC
Electricity	0.0066	−0.0001	0.0123	0.0154	CRC
Traffic	0.0124	−0.0519	0.0142	0.0226	CRC

Across 28 dataset–backbone pairs, CRC has lower averaged error in 18 cases (PIR: 9; tie: 1).

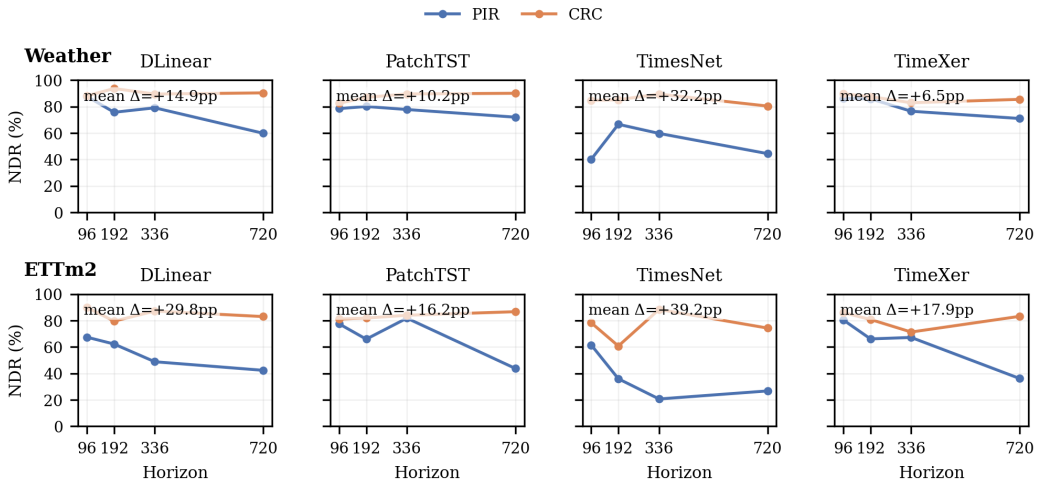

Figure 2: **NDR stability curves.** Non-degradation rate (NDR, %) of PIR vs. CRC across horizons on **Weather** and **ETTm2** (higher is better). Each panel corresponds to one backbone; annotations report mean ΔNDR in percentage points (pp). Exact values are provided in Appendix Table 5.

As shown in Figure 2, CRC yields consistently higher NDR than PIR across horizons on both datasets for most backbones, indicating fewer harmful updates under selective deployment. Notably, the mean ΔNDR improvements are largest on TimesNet, aligning with CRC's goal of suppressing correction-induced degradation.

4.4 Ablation Study: What Makes Selective Correction Work?

We zoom in on the DLinear–Electricity panel in Table 2. Firewall-style ablations appear as Group 2 in the same table; this subsection focuses on encoder and hybrid-head design (Groups 1 and 3). We report component-wise ablations on one backbone–dataset pair to control computational cost; Table 1 already varies datasets and backbones for end-to-end accuracy.

Structured representation matters. Removing the direction-aware encoder or replacing it with simpler features (Group 3) consistently weakens performance relative to the full model. This indicates that selective correction benefits from representations that expose structured residual patterns.

Conservative recovery and conditional refinement play complementary roles. Using an MLP-only corrector (Group 1a) leads to severe degradation, with NDR dropping sharply to 70.0%, showing that expressive refinement alone is too unstable. By contrast, a ridge-only corrector remains much safer but underpowered. These results are consistent with the conservative–conditional decomposition:

the ridge term is relatively stable, while nonlinear refinement helps most when paired with deployment constraints.

Overall, the ablation aligns with the design of CRC: structured residual evidence enables correction, conservative recovery stabilizes it, and the hybrid head is most effective under selective deployment. Encoder dynamics are illustrated in Appendix C (Figure 3); inference-time τ scaling and validation-ratio robustness are summarized in Section 4.2.5.

5 Discussion

Selective deployment for robust MSE gains. CRC focuses on gating when to apply post-hoc updates to a frozen forecaster. When residual evidence is weak, deploying a correction can increase test error even if the corrector fits the training data well. NDR is therefore used as an operational diagnostic of harmful-update frequency, rather than as a formal safety certificate.

Operational guidelines and monitoring signals. CRC exposes deployment-time diagnostics (e.g., clipping/abstention rates and selected-path frequencies) and simple knobs that trade off accuracy and stability (Section 4.2.5). A fail-safe path always remains: revert to the frozen backbone (or ridge floor only).

Relation to UQ/abstention and limitations. Conformal and selective prediction [22, 3] focus on coverage or deferral. CRC instead gates post-hoc point refinements to avoid local degradations, and can be combined with interval methods when needed. As a hold-out-calibrated procedure, CRC relies on the validation period being representative of deployment conditions; under substantial validation–test shift (or backbone changes), the frozen rule may require recalibration. On B2 in Table 3, the ridge-only gate keeps NDR high but MAE slightly above the frozen baseline, which is a concrete case where conservative gating is not free. Recalibration leaves the backbone fixed and refits only the post-hoc stage on a new validation slice.

6 Conclusion

We presented CRC, a plug-and-play framework for *robust residual correction via selective deployment* in multivariate forecasting. It combines a direction-aware encoder with a hybrid residual corrector and a validation-calibrated deployment firewall that stabilizes post-hoc refinement and improves test-time MSE in most evaluated settings. The main conclusion is *post-hoc correction should be selective, not universal*: robust MSE gains require controlling *when* updates deploy rather than applying residual regression indiscriminately. CRC operationalizes this principle through validation-calibrated gates.

Limitations include reliance on a fixed graph prior (we do not perform causal discovery), conservative gating that may suppress rare large improvements, and dependence of the frozen rule on the validation period, including under distribution shift between validation and test (while robustness to validation budget within the standard protocol is summarized in Section 4.2.5). Future work includes learned or adaptive graphs, stronger and pretrained backbones, and benchmarks with pronounced spikes or zero inflation to stress-test robustness mechanisms.

Acknowledgments and Disclosure of Funding

This work is supported by HKUST-GZ Startup Fund G0101000338.

The authors declare no competing interests.

References

[1] Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. *Advances in Neural Information Processing Systems*, 33:21865–21877, 2020.

[2] 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, pages 341–352, 2024.

[3] Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International conference on machine learning*, pages 2151–2159. PMLR, 2019.

[4] Hardik Goel, Igor Melnyk, and Arindam Banerjee. R2n2: Residual recurrent neural networks for multivariate time series forecasting. *arXiv preprint arXiv:1709.03159*, 2017.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[6] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[7] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[8] 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, pages 17868–17876, 2025.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Daojun Liang, Qi Li, Yinglong Wang, Jing Chen, Hu Zhang, Xiaoxiao Cui, Qizheng Wang, and Shuo Li. The forecast after the forecast: A post-processing shift in time series. *arXiv preprint arXiv:2601.20280*, 2026.

[11] 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.

[12] Zhiding Liu, Mingyue Cheng, Guanhao Zhao, Jiqian Yang, Qi Liu, and Enhong Chen. Improving time series forecasting via instance-aware post-hoc revision. *arXiv preprint arXiv:2505.23583*, 2025.

[13] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[14] Heitor R Medeiros, Hossein Sharifi-Noghabi, Gabriel L Oliveira, and Saghar Irandoust. Accurate parameter-efficient test-time adaptation for time series forecasting. *arXiv preprint arXiv:2506.23424*, 2025.

[15] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1):115–166, 2018.

[16] Pablo Montero-Manso, George Athanasopoulos, Rob J Hyndman, and Thiyanga S Talagala. Fforma: Feature-based forecast model averaging. *International Journal of Forecasting*, 36(1):86–92, 2020.

[17] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

[18] 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.

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[20] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. Detecting causal associations in large nonlinear time series datasets. *arXiv preprint arXiv:1702.07007*, 2017.

[21] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

[22] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of machine learning research*, 9(3), 2008.

[23] Kamile Stankeviciute, Ahmed M Alaa, and Mihaela Van der Schaar. Conformal time-series forecasting. *Advances in neural information processing systems*, 34:6216–6228, 2021.

[24] 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.

[25] Haochun Wang, Ruichen Xu, Georgios Kementzidis, Karen Cho, Sebastian Ramirez Villarreal, and Yuefan Deng. Towards principled test-time adaptation for time series forecasting. *arXiv preprint arXiv:2605.17250*, 2026.

[26] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. Timexer: Empowering transformers for time series forecasting with exogenous variables. *Advances in Neural Information Processing Systems*, 37:469–498, 2024.

[27] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

[28] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[29] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

[30] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

[31] Chen Xu and Yao Xie. Conformal prediction interval for dynamic time-series. In *International Conference on Machine Learning*, pages 11559–11569. PMLR, 2021.

[32] Chen Xu and Yao Xie. Sequential predictive conformal inference for time series. In *International Conference on Machine Learning*, pages 38707–38727. PMLR, 2023.

[33] 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, pages 11121–11128, 2023.

[34] 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 *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

A Theoretical Analysis and Proofs

This appendix collects design-oriented statements under explicit assumptions. They motivate why clipping and direction-consistent gating can limit harmful updates; they are not test-time certificates under distribution shift or arbitrary dependence across units. Proposition A.3 holds by construction when validation selects among candidates that include the baseline. Remark A.4 gives informal PAND motivation only and is *not* a valid finite-sample lower bound on test NDR. The main paper's primary empirical target remains reduced test MSE.

We first give pointwise and validation-level non-degradation statements under stated conditions, then informal probably approximately non-degrading (PAND) motivation (Remark A.4). Finally, we discuss when a direction-aware encoder can aid recovery of systematic residual structure.

A.1 Validation-Calibrated Selection Properties

Let Y denote the ground-truth future and $\widehat{Y}^{\mathrm{base}}$ the baseline forecast. Let Δ_{lin} denote the linear (ridge) correction. We define the residual after the linear floor as

$$e \;=\; Y - \big(\widehat{Y}^{\mathrm{base}} + \Delta_{\mathrm{lin}}\big).$$

Let δ denote the nonlinear correction proposal, and let the clipped nonlinear update be $\delta' = \mathrm{clip}(\delta, -\tau, \tau)$ with threshold $\tau > 0$.

Oracle condition vs. deployable rule. At test time, the true residual e is unobservable. Therefore, the condition below should be understood as an *oracle sufficient condition* for pointwise non-degradation. CRC does *not* assume test-time access to e; instead, it enforces a conservative *training/validation-time gating* using observable residuals on held-out data and *freezes* the resulting rule for deployment. The validation-level selection and shrink-to-base mechanisms (Prop. A.3) provide a deployable validation-calibrated selection property (the baseline remains an explicit candidate), without requiring test-time residual access.

Assumption A.1 (Directional gating and bounded update (oracle sufficient condition))**.** (Oracle condition.) The clipped nonlinear update is direction-aligned with the post-linear residual and bounded in magnitude, i.e.,

$$\mathrm{sign}(\delta') = \mathrm{sign}(e), \qquad 0 \le |\delta'| \le \min\{|e|, \tau\}.$$

Proposition A.2 (Pointwise non-degradation under MAE/MSE)**.** *Under Assumption A.1, the pointwise error does not increase after applying the clipped nonlinear update:*

$$\big|e - \delta'\big| \;\le\; |e|.$$

Moreover, the same condition implies pointwise non-increase of MSE:

$$(e - \delta')^2 \;\le\; e^2.$$

Proof. Under Assumption A.1, $\mathrm{sign}(\delta') = \mathrm{sign}(e)$ and $|\delta'| \le |e|$. Hence $|e - \delta'| = ||e| - |\delta'|| \le |e|$, proving the MAE claim. Squaring both sides yields $(e - \delta')^2 \le e^2$, proving the MSE claim. □

Proposition A.3 (Validation-level non-degradation w.r.t. the baseline)**.** *On the validation set, for each evaluation unit (e.g., each (i, h) pair under the NDR definition), select the best-performing predictor among*

$$\widehat{Y}^{\mathrm{base}}, \qquad \widehat{Y}^{\mathrm{base}} + \Delta_{\mathrm{lin}}, \qquad \widehat{Y}^{\mathrm{base}} + \Delta_{\mathrm{lin}} + \delta',$$

according to the validation loss, and freeze this choice for deployment. Then the validation error satisfies

$$\mathcal{L}_{\mathrm{val}}(\mathrm{CRC}) \;\le\; \mathcal{L}_{\mathrm{val}}(\mathrm{base}),$$

and consequently $\mathrm{NDR}_{\mathrm{val}} = 100\%$ at the same evaluation granularity. If a positive shrink-to-base margin $\varepsilon > 0$ is used, some units may default to the baseline when the best CRC path beats the baseline by less than ε on validation; the baseline remains an explicit candidate, so per-unit validation loss is still no worse than that of the baseline.

Proof. For each evaluation unit, the selection rule explicitly chooses the candidate with the smallest validation loss from a set that includes the baseline itself. Therefore, the deployed predictor cannot incur higher validation loss than the baseline on any unit. Aggregating over all units yields the stated inequality. The NDR result follows directly from its definition. □

Remark A.4 (Informal PAND motivation; not a test-set guarantee). Let m denote the number of validation evaluation units and define $Z_k = \mathbb{1}\left\{\ell(\widehat{Y}^{\mathrm{CRC}}, Y) \le \ell(\widehat{Y}^{\mathrm{base}}, Y)\right\}$ at unit k, where $\ell(\cdot,\cdot)$ is the per-unit loss used in NDR (e.g., MAE or MSE), with $\mathrm{NDR}_{\mathrm{val}} = \frac{1}{m}\sum_{k=1}^{m} Z_k$. High $\mathrm{NDR}_{\mathrm{val}}$ is *encouraging* only under idealized conditions: the deployment rule is treated as fixed when judging test units, and units behave like i.i.d. draws from a stable per-unit success law. In practice, the rule is fit and selected using validation data (selection bias), units across (i,h) may be dependent, and a rigorous validation-to-test link requires separate concentration for validation *and* test empirical means—not a one-step corollary of Hoeffding [6] on validation alone. We therefore do *not* state a finite-sample inequality of the form $\mathrm{NDR}_{\mathrm{test}} \ge \mathrm{NDR}_{\mathrm{val}} - \sqrt{\log(2/\alpha)/(2m)}$; PAND language is qualitative motivation to read alongside reported test NDR and the sensitivity analyses in Appendix C (Section 4.2.5), not a deployment certificate under shift.

A.2 When Does the Direction-Aware Encoder Help Recovery?

Consider the following data-generating process for node i:

$$Y_{:,i} \;=\; f_i(W_{:,i}) \;+\; \sum_{j\in \mathrm{Pa}(i)} g_{j\to i}(W_{:,j}) \;+\; \varepsilon_i, \tag{14}$$

where the baseline $\widehat{Y}_i^{\mathrm{base}}$ already captures $f_i(\cdot)$ but may miss the cross-node systematic term. Define the baseline residual

$$r_i \;=\; Y_{:,i} - \widehat{Y}_i^{\mathrm{base}}.$$

Let the encoder output be

$$Z_i \;=\; \alpha_{i\to i} U_i \;+\; \sum_{j\in\mathcal{N}(i)} \alpha_{j\to i} U_j,$$

where U_j denotes node-wise latent features extracted from historical windows, and $\mathcal{N}(i)$ covers $\mathrm{Pa}(i)$.

Remark A.5 (Ridge floor as design motivation (not a recovery guarantee)). When cross-node effects admit a local linearization and the encoder features Z_i approximately span the associated residual subspace, fitting a ridge map on Z_i provides a ridge-regularized linear approximation to the residual—a conservative correction floor rather than a certified recovery of systematic error. Embedding distortion, noise, and model misspecification affect this approximation in practice. The encoder supplies direction-aware inductive bias rather than causal discovery; nonlinear refinement and the validation-calibrated deployment rule guard updates beyond this floor.

B Implementation Details

B.1 Encoder, priors, and directional gates (compact)

Multi-scale priors. We augment the learned encoder output with fixed per-node priors $\mathbf{F}$ computed over multiple temporal scales: basic distribution statistics (mean/variance/IQR), temporal dynamics (EMA and absolute-difference energy), correlation features (multi-scale lagged auto-correlation), and frequency features (band power via rFFT). Priors are aggregated through the row-normalized adjacency $\mathbf{A}_{\mathrm{norm}}$ and serve as explicit inductive bias under non-stationarity.

Pairwise interaction module. The shared ordered-pair module f_{pair} consumes windows $(W_{:,:,i}, W_{:,:,j})$ and outputs an interaction embedding plus a directional gate. Internally it produces a 2×2 matrix $\mathbf{M}_{ij}$; the off-diagonal entry corresponding to $j\to i$ is passed through $\tanh$ to form $\alpha_{j\to i}$ (directional evidence, not causal identification).

B.2 Training Hyperparameters

CRC is implemented in PyTorch [19] and trained on NVIDIA GPUs using a two-stage procedure: (i) multi-task pre-training of the encoder, then (ii) fitting the conservative–conditional corrector. We use AdamW [13] (encoder) and Adam [9] (MLP) with learning rate 10^{-4}, batch size 32, and early stopping on validation MAE. The graph prior is a correlation-based K-NN adjacency built on the training split with K=5 and then fixed. Unless stated otherwise, quantile clipping uses Q=0.80 and shrink-to-base margin uses ε=0.01. All post-hoc modules are tuned on the standard train/val folds using frozen backbone checkpoints.

B.3 Deployment Rule Instantiation

CRC deploys a frozen, validation-calibrated choice among a small candidate set at the NDR granularity (per (i, h) unit). For each unit we select the lowest validation-loss predictor from $\{\widehat{Y}^{\text{base}},\ \widehat{Y}^{\text{base}} + \Delta_{\text{lin}},\ \widehat{Y}^{\text{base}} + \Delta_{\text{lin}} + \delta'\}$, and freeze the selection for test. Equivalently, $\widehat{Y} = \widehat{Y}^{\text{base}} + w_1\Delta^{\text{base}} + w_2\Delta^{\text{clip}}$ with $(w_1, w_2) \in \{(0,0),(1,0),(1,1)\}$ chosen on validation. The clipping threshold τ is estimated from validation via quantile clipping (default Q=0.80), and shrink-to-base reverts to the baseline unless the best candidate improves validation loss by at least ε=0.01.

B.4 Computational Complexity

The ridge term admits a closed-form solution, the MLP is small, and the encoder operates on fixed windows. Overall complexity is $O(BPN + N^2d)$ with embedding dimension d, batch B, window P, and N variables; the N^2d term reflects the pairwise interaction blocks (Section 3.2), consistent with the $O(N^2)$ scaling discussed in the main text. This makes CRC efficient to train and easy to plug into diverse backbones.

C Supplementary Experimental Analyses

C.1 Detailed test-time NDR vs. PIR (Weather)

Table 5 provides a horizon-wise **NDR** comparison supporting the main-text PIR vs. CRC results.

Table 5: Non-Degradation Rate (NDR, %) comparison between PIR and CRC on **Weather** (left) and **ETTm2** (right) across backbones and forecasting horizons. Higher is better.

Weather

Backbone	Horizon	PIR	CRC
DLinear	96	87.7	**88.2**
	192	75.7	**93.7**
	336	79.1	**89.5**
	720	59.8	**90.4**
PatchTST	96	78.5	**82.5**
	192	80.1	**87.0**
	336	77.8	**89.6**
	720	72.0	**90.1**
TimesNet	96	40.1	**84.7**
	192	66.6	**85.2**
	336	59.7	**89.5**
	720	44.4	**80.4**
TimeXer	96	86.3	**89.7**
	192	85.9	**87.6**
	336	76.5	**82.9**
	720	71.0	**85.5**

ETTm2

Backbone	Horizon	PIR	CRC
DLinear	96	67.4	**90.0**
	192	62.2	**79.5**
	336	48.8	**87.2**
	720	42.3	**83.0**
PatchTST	96	77.5	**81.1**
	192	65.9	**81.9**
	336	81.8	**84.0**
	720	43.7	**86.7**
TimesNet	96	61.5	**78.3**
	192	35.9	**60.7**
	336	20.6	**88.4**
	720	26.7	**74.3**
TimeXer	96	80.4	**86.0**
	192	66.1	**80.8**
	336	67.2	**71.3**
	720	36.1	**83.2**

C.2 Temporal evolution of learned directional influences

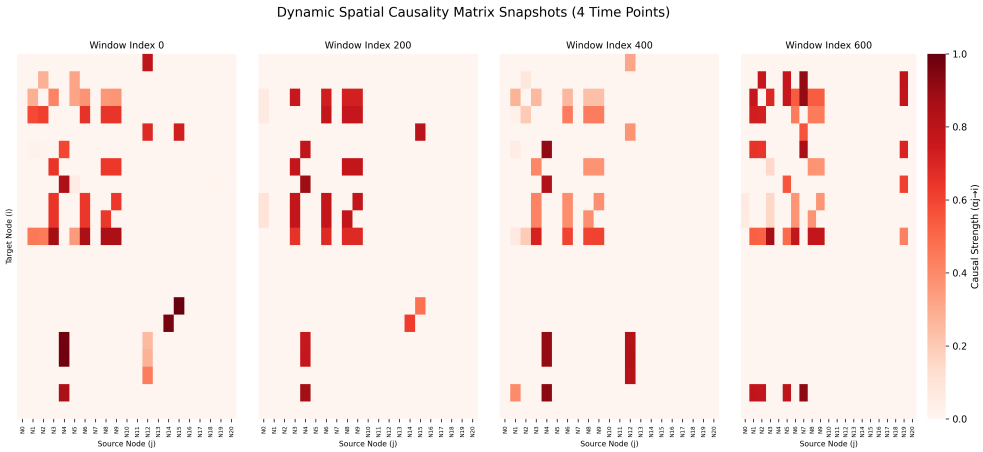

Figure 3: **Temporal evolution of learned directional interaction scores (N=4).** Instantaneous directional scores $S_t[i, j]$ (messaging $j \to i$) with a **TimesNet** backbone on **Weather**. Four windows ($\Delta t = 200$) illustrate non-stationary dependencies used as residual evidence; these scores provide directional evidence and are *not* interpreted as identified causal effects.

C.3 Sensitivity to clipping and validation calibration

We examine robustness of the MSE–NDR tradeoff to (i) rescaling the validation-calibrated clipping threshold τ and (ii) perturbing validation-side calibration of the frozen intervention rule (main-text discussion: Section 4.2.5). Figures 4 and 5 report the resulting accuracy–reliability tradeoffs and their sensitivity to the amount of validation data used for calibration, respectively.

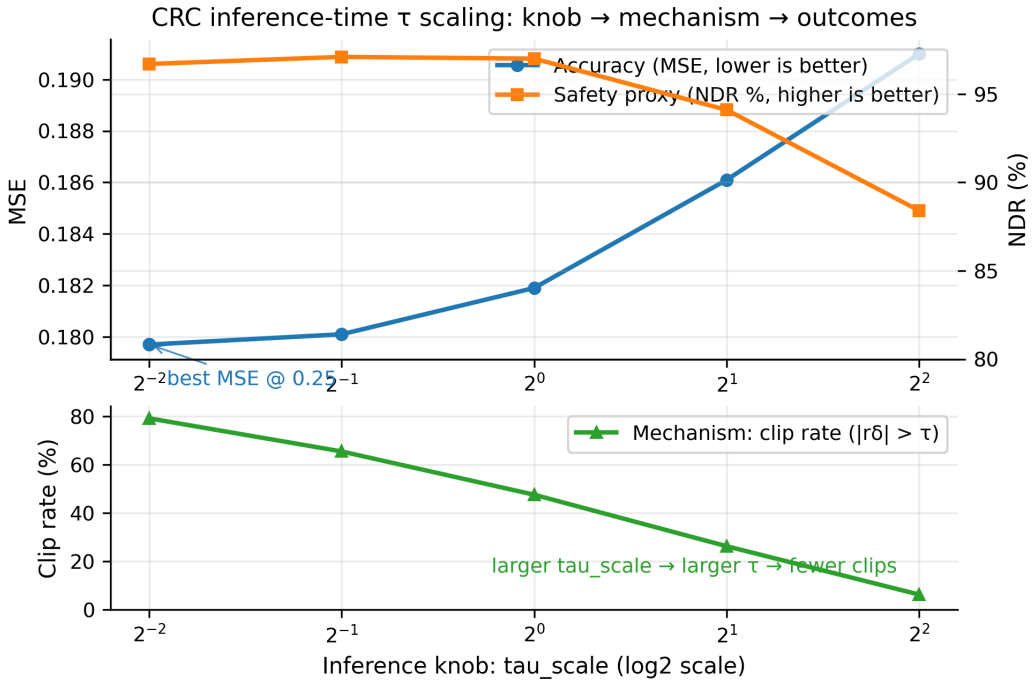

Figure 4: MSE–NDR tradeoff under rescaling of the calibrated quantile-clipping threshold τ.

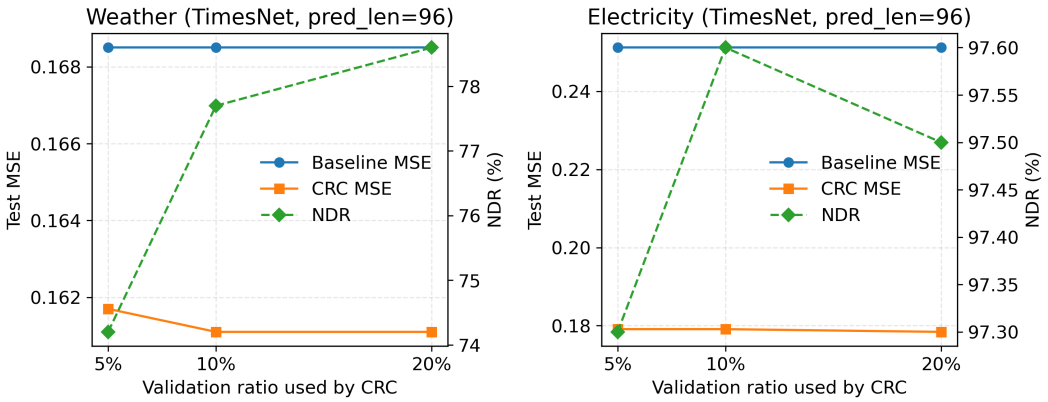

Figure 5: Sensitivity of the MSE–NDR profile to validation calibration of the frozen intervention rule.

D Controlled Deployment Ladder and Seed Variability

D.1 Notes on the Controlled Deployment Ladder

The full G0–G5 controlled deployment ladder is reported in Table 3 (Section 4.2). G1 (always-deploy under the protocol) drops NDR sharply relative to G5; G2 can stay safe yet under-improve (B2 MAE above baseline despite high NDR). Main-table baseline/CRC rows match G0/G5 only as policies: those cells are horizon-averaged and need not equal the single-horizon numbers in Table 3.

D.2 Representative seed-variability analysis (B1 only)

Table 6: Representative seed-variability analysis (B1 only). ETTh1, DLinear, pred_len=96 (B1). Mean $\pm$ std over training seeds $\{2021, 2022, 2023\}$ for the post-hoc CRC module (frozen backbone). This table reports an independent three-seed robustness evaluation on the representative B1 setting, complementing the headline main-table results (Table 1). Table 3 is an exception (controlled deployment ladder, seed 2021 only): G5 B1 MAE 0.3914 vs. seed-averaged 0.3991—both are valid under different reporting protocols.

Config	MAE	MSE	NDR (%)
G5 Full CRC	0.3991 ± 0.0011	0.3785 ± 0.0007	79.3 ± 6.3
G1 Always-on	0.4828 ± 0.0131	0.4936 ± 0.0378	7.2 ± 1.5

E Additional Experimental Results

This section presents additional experimental results that complement the summary statistics reported in the main text by providing horizon-wise results underlying Table 4. All comparisons use identical backbone checkpoints, splits, and post-hoc fitting protocols. Table 7 reports absolute errors (with/without CRC), while Table 8 reports PIR$-$ CRC differences (positive favors CRC), exposing horizon-specific behavior that can be obscured by horizon-averaged metrics.

Table 7: Results on ETTh1, ETTh2, ETTm1, ETTm2, Weather, Electricity, and Traffic datasets (horizons 96/192/336/720). Lower is better. Bold indicates that CRC outperforms the corresponding baseline for that metric.

Dataset	Horizon	TimeXer		+ CRC		TimesNet		+ CRC		PatchTST		+ CRC		DLinear		+ CRC	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.3818	0.4029	**0.3790**	**0.3989**	0.3984	0.4163	0.3986	**0.4152**	0.3792	0.3988	**0.3790**	**0.3898**	0.3962	0.4108	**0.3873**	**0.4099**
	192	0.4285	0.4355	**0.4100**	**0.4341**	0.4373	0.4399	0.4473	**0.4374**	0.4240	0.4304	**0.4218**	**0.4252**	0.4450	0.4404	**0.4392**	**0.4357**
	336	0.4672	0.4494	0.4704	0.4500	0.4722	0.4584	**0.4650**	0.4586	0.4686	0.4578	**0.4663**	0.4651	0.4874	0.4654	**0.4820**	0.4655
	720	0.4988	0.4926	**0.4967**	0.4931	0.5181	0.4965	**0.5140**	0.4989	0.5193	0.5036	0.5200	**0.5015**	0.5126	0.5104	0.5242	0.5106
ETTh2	96	0.2854	0.3376	0.2858	**0.3372**	0.3423	0.3792	**0.3373**	**0.3748**	0.2924	0.3468	**0.2906**	**0.3446**	0.3414	0.3953	**0.3172**	**0.3663**
	192	0.3628	0.3892	**0.3613**	**0.3885**	0.4068	0.4117	**0.4060**	**0.4109**	0.3794	0.4044	**0.3761**	**0.4007**	0.4818	0.4792	**0.4499**	**0.4504**
	336	0.4109	0.4222	0.4132	0.4244	0.4398	0.4421	**0.4382**	0.4436	0.4183	0.4325	**0.4116**	**0.4290**	0.5929	0.5422	**0.5773**	**0.5333**
	720	0.4485	0.4534	0.4492	0.4561	0.4440	0.4564	0.4512	0.4584	0.4328	0.4532	0.4392	0.4567	0.8403	0.6611	**0.8089**	**0.6381**
ETTm1	96	0.3178	0.3563	**0.3173**	**0.3544**	0.3444	0.3806	**0.3362**	**0.3770**	0.3268	0.3667	**0.3209**	**0.3617**	0.3459	0.3737	**0.3321**	**0.3664**
	192	0.3616	0.3829	**0.3600**	**0.3818**	0.4222	0.4142	**0.4051**	**0.4100**	0.3718	0.3917	**0.3694**	**0.3897**	0.3818	0.3911	**0.3704**	**0.3859**
	336	0.3951	0.4067	0.3996	0.4085	0.4251	0.4276	**0.4242**	**0.4269**	0.3981	0.4080	**0.3932**	**0.4050**	0.4153	0.4150	**0.4043**	**0.4125**
	720	0.4524	0.4413	0.4616	0.4480	0.4879	0.4610	0.4911	0.4640	0.4586	0.4451	0.4618	0.4513	0.4730	0.4509	**0.4672**	0.4553
ETTm2	96	0.1706	0.2555	**0.1701**	**0.2553**	0.1858	0.2667	**0.1842**	0.2672	0.1868	0.2692	**0.1849**	**0.2675**	0.1934	0.2928	**0.1836**	**0.2744**
	192	0.2365	0.2987	**0.2300**	**0.2961**	0.2532	0.3100	**0.2486**	**0.3095**	0.2482	0.3070	**0.2453**	**0.3046**	0.2845	0.3614	**0.2647**	**0.3364**
	336	0.2954	0.3380	**0.2874**	**0.3365**	0.3237	0.3496	0.3240	0.3502	0.3109	0.3492	**0.3069**	**0.3476**	0.3849	0.4295	**0.3479**	**0.3989**
	720	0.3916	0.3935	**0.3876**	**0.3921**	0.4162	0.4050	**0.4102**	**0.4035**	0.4233	0.4157	**0.4182**	**0.4123**	0.5561	0.5235	**0.5537**	**0.5137**
Weather	96	0.1574	0.2047	**0.1545**	**0.2018**	0.1684	0.2185	**0.1622**	**0.2120**	0.1752	0.2174	**0.1616**	**0.2071**	0.1962	0.2561	**0.1635**	**0.2158**
	192	0.2041	0.2475	**0.2008**	**0.2450**	0.2345	0.2735	**0.2069**	**0.2538**	0.2214	0.2562	**0.2043**	**0.2465**	0.2389	0.2992	**0.2086**	**0.2631**
	336	0.2610	0.2899	**0.2530**	**0.2852**	0.2902	0.3095	**0.2682**	**0.2965**	0.2801	0.2975	**0.2531**	**0.2847**	0.2811	0.3306	**0.2543**	**0.3004**
	720	0.3400	0.3406	**0.3189**	**0.3283**	0.3593	0.3539	**0.3230**	**0.3360**	0.3560	0.3475	**0.3257**	**0.3319**	0.3454	0.3819	**0.3277**	**0.3540**
Electricity	96	0.1406	0.2426	**0.1345**	**0.2326**	0.1676	0.2702	**0.1564**	**0.2591**	0.1802	0.2734	**0.1509**	**0.2481**	0.2102	0.3014	**0.1516**	**0.2517**
	192	0.1572	0.2558	**0.1517**	**0.2485**	0.1897	0.2914	**0.1762**	**0.2787**	0.1875	0.2797	**0.1639**	**0.2586**	0.1938	0.2799	**0.1722**	**0.2649**
	336	0.1759	0.2754	**0.1682**	**0.2666**	0.1981	0.2988	**0.1898**	**0.2916**	0.2042	0.2959	**0.1800**	**0.2762**	0.2230	0.3191	**0.1833**	**0.2836**
	720	0.2161	0.3091	**0.1990**	**0.2957**	0.2313	0.3249	**0.2232**	**0.3213**	0.2435	0.3281	**0.2142**	**0.3060**	0.2440	0.3323	**0.2132**	**0.3099**
Traffic	96	0.4277	0.2715	**0.4221**	**0.2641**	0.5913	0.3160	**0.5571**	**0.2972**	0.4583	0.2985	**0.4298**	**0.2808**	0.6965	0.4287	**0.4889**	**0.3084**
	192	0.4587	0.2761	**0.4558**	**0.2698**	0.6120	0.3278	**0.5847**	**0.3059**	0.4804	0.3104	**0.4543**	**0.2919**	0.6014	0.3747	**0.4839**	**0.3083**
	336	0.4762	0.2925	**0.4699**	**0.2847**	0.6543	0.3484	**0.6192**	**0.3183**	0.4963	0.3174	**0.4698**	**0.2990**	0.6077	0.3769	**0.5036**	**0.3187**
	720	0.5195	0.3101	**0.5135**	**0.3031**	0.6711	0.3490	**0.6546**	**0.3339**	0.5308	0.3341	**0.5044**	**0.3174**	0.6482	0.3982	**0.5404**	**0.3383**

Table 8: Relative performance of **PIR (Post-forecasting Identification and Revision)** vs. **CRC** ($\Delta = \text{PIR} - \text{CRC}$) across datasets, horizons, and backbones. Negative values indicate PIR better; positive values indicate CRC better.

Dataset	Horizon	TimeXer		TimesNet		PatchTST		DLinear	
		ΔMSE	ΔMAE	ΔMSE	ΔMAE	ΔMSE	ΔMAE	ΔMSE	ΔMAE
ETTh1	96	−0.0030	−0.0019	−0.0076	−0.0002	−0.0033	0.0012	−0.0143	−0.0159
	192	−0.0060	−0.0041	0.0057	0.0086	0.0072	0.0038	−0.0152	−0.0227
	336	−0.0264	−0.0213	0.0150	0.0104	0.0077	−0.0011	−0.0100	−0.0211
	720	−0.0537	−0.0391	−0.0130	−0.0019	−0.0260	−0.0245	−0.0472	−0.0485
ETTh2	96	0.0092	0.0048	0.0097	0.0012	0.0134	0.0064	−0.0312	−0.0313
	192	0.0057	0.0045	0.0160	0.0072	0.0059	0.0013	−0.0819	−0.0634
	336	0.0038	0.0016	0.0118	0.0024	0.0464	0.0210	−0.1643	−0.1093
	720	−0.0492	−0.0311	−0.0162	−0.0114	−0.0092	−0.0067	−0.3919	−0.2011
ETTm1	96	0.0017	0.0016	0.0278	0.0050	0.0031	−0.0007	−0.0161	−0.0134
	192	0.0070	0.0042	−0.0251	−0.0130	−0.0004	−0.0017	−0.0034	−0.0049
	336	0.0244	0.0155	0.0198	0.0101	0.0108	0.0080	−0.0053	−0.0065
	720	0.0134	0.0100	0.0239	0.0100	0.0032	0.0017	−0.0012	−0.0133
ETTm2	96	0.0089	0.0057	0.0018	0.0018	0.0101	0.0075	−0.0056	−0.0104
	192	0.0100	0.0079	0.0234	0.0095	0.0097	0.0134	−0.0217	−0.0304
	336	0.0196	0.0065	0.0170	0.0088	0.0081	0.0034	−0.0399	−0.0529
	720	0.0434	0.0249	0.0128	0.0095	0.0128	0.0107	−0.1477	−0.1087
Weather	96	0.0025	0.0032	0.0058	0.0060	0.0104	0.0069	0.0185	0.0112
	192	0.0052	0.0040	0.0141	0.0072	0.0147	0.0105	0.0194	−0.0001
	336	0.0110	0.0078	0.0148	0.0085	0.0249	0.0133	0.0287	0.0016
	720	0.0281	0.0197	0.0420	0.0250	0.0983	0.0461	0.0403	0.0030
Electricity	96	0.0125	0.0084	0.0016	0.0019	0.0151	0.0009	0.0174	−0.0017
	192	0.0083	0.0045	0.0038	0.0003	0.0121	0.0014	0.0068	−0.0039
	336	0.0058	0.0034	−0.0018	−0.0046	0.0120	−0.0012	0.0127	−0.0056
	720	0.0000	−0.0007	−0.0042	−0.0053	0.0098	−0.0010	0.0248	0.0041
Traffic	96	0.0139	0.0059	−0.0441	0.0068	0.0202	0.0022	0.0201	0.0016
	192	0.0072	0.0082	−0.0447	0.0081	0.0127	−0.0049	0.0241	0.0057
	336	0.0101	0.0003	−0.0622	0.0127	0.0142	−0.0060	0.0224	−0.0047
	720	0.0185	0.0019	−0.0566	0.0071	0.0096	−0.0064	0.0236	−0.0073

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state that CRC combines a hybrid residual corrector with validation-calibrated selective deployment, report evaluation on seven standard benchmarks and four backbones, and scope claims (test MSE under the benchmark protocol; no distribution-shift guarantees) consistent with Section 5, Section 6, and Appendix A.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 and Section 6 discuss reliance on a fixed graph prior, conservative gating, validation-period dependence (including shift), and Appendix A clarifies that theoretical statements are conditional and not unconditional deployment certificates.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Appendix A states assumptions alongside Propositions A.2 and A.3 with complete proofs; Remark A.5 summarizes ridge-floor motivation without a finite-sample guarantee.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 4 specifies datasets, horizons, metrics (including NDR), backbones, and the evaluation protocol; Section 3 and Appendix B.2, B.3 describe the training stages, optimizers, key hyperparameters (K, Q, ε), and the frozen deployment rule, enabling replication with public benchmarks and cited backbone setups.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Experiments use public time-series benchmarks cited in the paper, and the abstract points to our code repository at `https://github.com/Xavian1-oss/CRC.git` with implementation and reproduction instructions.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 describes datasets, standard long-horizon splits, metrics, and seed averaging; Appendix B.2 and B.3 give optimizer choices, learning rates, batch size, early stopping, adjacency construction, and deployment-rule instantiation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Section 4 reports headline metrics averaged over three independent random seeds (sample mean; not every table includes error bars). Appendix D.2 additionally reports mean $\pm$ standard deviation on a representative setting, and Section 4.2.5 with Appendix C report sensitivity of outcomes to deployment hyperparameters; NDR complements pooled errors as a harmful-update diagnostic.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix B.2 documents a PyTorch implementation trained on NVIDIA GPUs with concrete training settings (optimizers, batch size, early stopping); Appendix B.4 summarizes asymptotic scaling with batch, horizon/window, and graph size. The code repository linked in the abstract specifies the full executable workload, from which wall-clock and memory footprints follow under standard single-/multi-GPU setups without needing per benchmark–backbone tables in the PDF.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses publicly available benchmark datasets and standard ML experimentation without human subjects, deceptive practices, or other activities inconsistent with the Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 1 ties forecasting to societal-scale decisions (transportation, energy, finance, urban operations) and notes that errors matter when forecasts gate operations or risk assessment; the granularity discussion warns that aggregate gains can coexist with localized degradations and uneven costs, while Section 5 emphasizes harmful-update risks under weak evidence and reliability limits under validation–test shift—together outlining benefits of improved deployment decisions and deployment-time failure modes relevant to real use.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The submission introduces a forecasting module evaluated on standard benchmarks and does not release a high-risk generative model, large scraped dataset, or analogous dual-use artifact requiring special release safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The benchmarks and backbone checkpoints follow publicly established forecasting suites and are credited via the cited dataset and method references; our implementation (`https://github.com/Xavian1-oss/CRC.git`) is released for reproducibility and we respect the terms associated with those public assets as documented by their sources.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: CRC implementation code is released via the repository linked in the abstract (`https://github.com/Xavian1-oss/CRC.git`), with accompanying documentation in the repository for training and evaluation on the cited benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or human-subjects studies.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subjects research is conducted.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Large language models are not used as part of the proposed forecasting methodology; any optional writing or editing assistance does not constitute a core methodological component per the NeurIPS policy.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.