SHIFT: PROGRESSIVE FEEDBACK FOR TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon forecasting is conventionally formulated as a one-shot prediction problem, where the entire future horizon is predicted solely from historical observations. We introduce **Progressive Feedback Forecasting**, a new forecasting paradigm in which future predictions are progressively committed, while newly revealed observations within the forecast horizon are causally incorporated to revise the still-unseen future. Under this paradigm, we propose **SHIFT**, a framework that maintains multiple complete future hypotheses together with a belief distribution over them. A history-conditioned generator constructs diverse hypotheses around a base forecast, while progressive feedback drives two complementary updates: a causal filter adjusts the relative belief of each hypothesis according to its consistency with revealed observations, and a recurrent assimilator revises their unreleased trajectories. To supervise intermediate belief updates, whose correct targets are unavailable causally, SHIFT introduces hindsight-teacher distillation, using full-future information only during training while preserving strictly causal inference. Experiments on diverse real-world benchmarks show that SHIFT reduces MSE and MAE by an average of **25.2%** and **15.1%**, respectively, over the strongest competing baseline on each dataset.

1 INTRODUCTION

Modern long-horizon forecasters typically follow a *one-shot* prediction paradigm, producing the entire future trajectory at once from a fixed historical context under point-wise objectives such as squared error (Wang et al., 2026b; Nie et al., 2023; Liu et al., 2024; Hu et al., 2026; Huang et al., 2026; Liu et al., 2026b). When several plausible futures are consistent with the observed history, such as demand that either spikes or stays flat, or a front that either arrives or stalls, a single point forecast may fall between incompatible futures rather than correspond to any plausible outcome (Bishop, 1994; Bhattacharyya et al., 2018). More expressive architectures can better exploit historical observations, but they remain fundamentally one-shot: once the forecast is issued, newly revealed observations within the forecast horizon are not used to revise the still-unseen future.

In many operational settings, forecasts are progressively consumed while new observations naturally become available before the entire forecast horizon has elapsed. This creates an inherent opportunity: partial realizations can provide direct evidence for revising the still-unseen future. Based on this observation, we formulate **Progressive Feedback Forecasting (PFF)**, a forecasting paradigm in which predictions are progressively committed and newly revealed observations within the forecast horizon are incorporated to refine subsequent predictions, as shown in Figure 1. Unlike conventional one-shot forecasting, PFF treats forecasting as an evolving process rather than a single irreversible decision. As feedback accumulates, it helps distinguish among initially plausible futures and progressively refines the remaining trajectory.

Turning this paradigm into a trainable system raises three coupled challenges. First, the model must represent multiple plausible futures rather than collapse them into a single trajectory. Second, newly revealed observations should not only help identify which future is more consistent with the unfolding sequence, but also revise the remaining shape of candidate trajectories when unexpected changes occur. Third, supervising such progressive updates is difficult because the unrevealed future needed to assess an intermediate prediction is not yet available during deployment. These challenges

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.

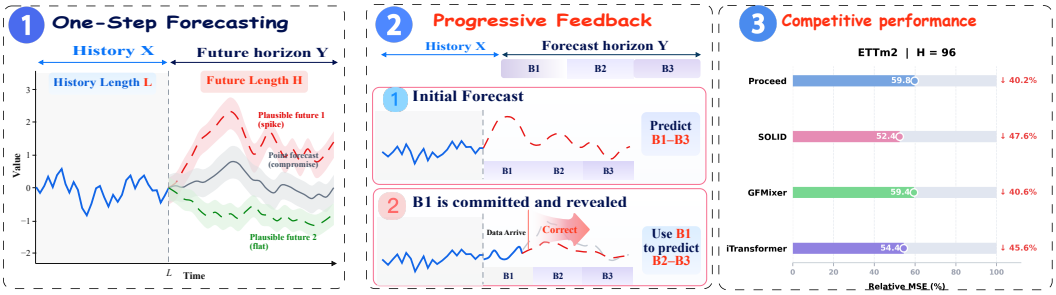

Figure 1: (a) One-shot forecasting and (b) our progressive feedback forecasting, along with (c) the competitive performance.

call for a framework that can maintain diverse future hypotheses, progressively refine them with incoming feedback, and learn reliable intermediate updates.

SHIFT (Scenario Hypothesis filtering with hIndsight-distilled Feedback Tracking) resolves this supervision problem through an asymmetric-information training scheme. During training, a *hindsight teacher* observes the complete target and forms the posterior that the full future ultimately supports; the causal student, which sees only revealed blocks, is distilled toward this posterior at every stage. This follows the broader principle of learning with privileged information and knowledge distillation (Vapnik & Vashist, 2009; Lopez-Paz et al., 2016; Hinton et al., 2015), but applies it to an evolving belief over future hypotheses rather than a fixed label. The remaining machinery makes the distilled filter operational: a scenario generator constructs hypotheses as smooth low-rank residual fields around a base forecast; a discounted log-linear update combines likelihood evidence with a learned calibrator; and a recurrent assimilator converts each revealed block into hypothesis-specific latent states that revise only the unreleased future.

Our contributions are fourfold. (i) We formulate time series forecasting with progressive feedback, where newly revealed observations within the forecast horizon can be causally exploited to revise the still-unobserved future. (ii) We propose **SHIFT**, which represents the future as a learned bank of hypotheses and jointly updates both their relative beliefs and unreleased trajectories as new evidence arrives, enabling forecast revision beyond correcting a single trajectory. (iii) We introduce *hindsight-teacher distillation* to supervise these otherwise ambiguous intermediate belief updates, using privileged full-future information during training while preserving strictly causal inference. (iv) Extensive experiments on multiple real-world benchmarks demonstrate the effectiveness and robustness of the proposed framework.

2 Related Work

Time series forecasting. Deep learning has substantially advanced time series forecasting, particularly for long-horizon prediction. Representative studies have explored lightweight temporal decomposition, patch-based representations, and explicit modeling of cross-variable dependencies (Zeng et al., 2023; Nie et al., 2023; Liu et al., 2024). More recent research increasingly emphasizes inductive biases tailored to the intrinsic structure of time series rather than simply increasing model complexity. In particular, frequency-aware and periodic representations have been developed to better characterize heterogeneous temporal patterns (Huang et al., 2025; Liu, 2025; Niu et al., 2026), while multiscale decomposition and global–local temporal modeling further improve the extraction of temporal dependencies across different resolutions and ranges (Hu et al., 2025; Cao et al., 2026). Overall, these advances have progressively strengthened how historical observations are represented and exploited for forecasting. Nevertheless, the prevailing formulation relies on a fixed historical context, largely overlooking forecast updates as new observations arrive within the horizon.

Online time series forecasting. Online time series forecasting extends conventional forecasting to continuously evolving data streams, where newly observed samples are used to adapt the forecaster during deployment. Existing studies mainly address distribution shifts and concept drift by enabling rapid adaptation while retaining previously acquired knowledge. Early approaches explore fast–slow adaptation and memory mechanisms (Pham et al., 2023), while subsequent methods improve adaptability through dynamically combining complementary forecasters (Zhang et al., 2023).

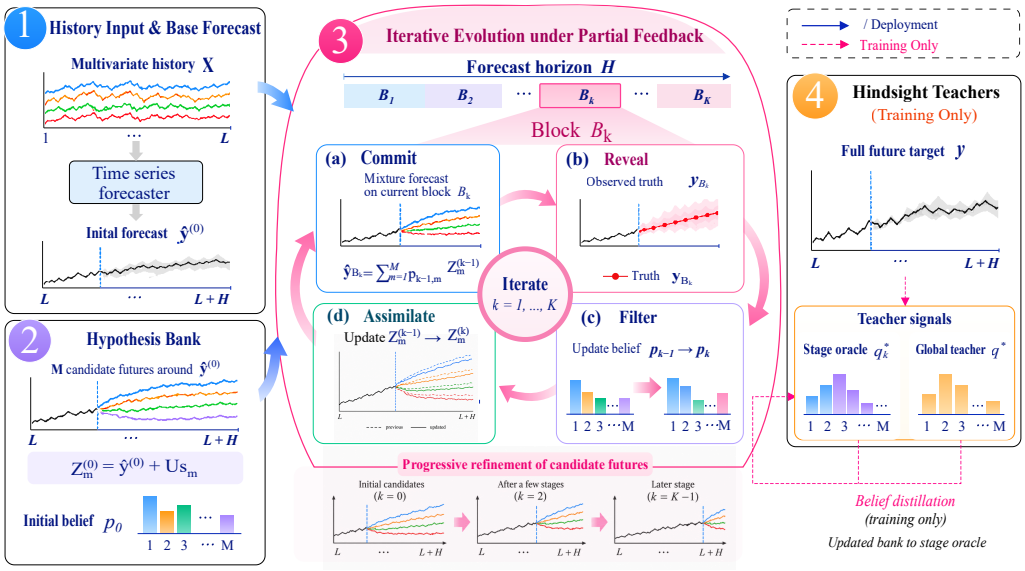

Figure 2: The overall framework of **SHIFT**.

Recent studies have paid increasing attention to more realistic online forecasting settings, including preventing information leakage when future labels arrive with delay (Lau et al., 2025), adapting pre-trained forecasters with partially observed ground truth (Kim et al., 2025), and proactively correcting models according to estimated distribution shifts (Zhao & Shen, 2025). More recently, feature-space adaptation has been explored as an alternative to directly updating forecasting parameters under delayed feedback (Huang et al., 2026). These methods primarily adapt a forecasting model across a sequential data stream to cope with evolving distributions. In contrast, our setting focuses on revision *within a single long-horizon forecasting episode*: newly revealed observations from the current horizon are used to revise the still-unobserved future, without requiring online parameter updates.

3 Problem Formulation

Time series forecasting. Given a multivariate historical sequence $\boldsymbol{X} \in \mathbb{R}^{L\times N}$, where L denotes the input length and N the number of variables, the goal of time series forecasting is to predict the future sequence $\boldsymbol{Y} \in \mathbb{R}^{H\times N}$, where H denotes the prediction horizon. In conventional long-horizon forecasting, the entire future sequence is predicted at once based solely on the observed history $\boldsymbol{X}$.

Progressive Feedback Forecasting. Unlike conventional forecasting, PFF divides the prediction horizon into $K = \lceil H/S \rceil$ consecutive blocks $\{\mathcal{B}_k\}_{k=1}^{K}$, where S denotes the block length. At stage k, the forecaster predicts the current block using the original history together with all previously revealed future observations,

$$\tilde{\boldsymbol{Y}}_{\mathcal{B}_k} = F\big(\boldsymbol{X}, \boldsymbol{Y}_{\mathcal{B}_{<k}}\big),$$

where $\boldsymbol{Y}_{\mathcal{B}_{<k}}$ denotes the observations revealed before stage k. Once the prediction for $\mathcal{B}_k$ has been committed, its ground truth $\boldsymbol{Y}_{\mathcal{B}_k}$ is revealed and becomes available for predicting subsequent blocks. Therefore, each prediction can exploit newly observed information from earlier stages, while the target block itself remains strictly unavailable at prediction time. The objective is to minimize the forecasting error over all stage-wise committed predictions: $\mathcal{L} = \frac{1}{HN}\sum_{k=1}^{K}\left\|\tilde{\boldsymbol{Y}}_{\mathcal{B}_k} - \boldsymbol{Y}_{\mathcal{B}_k}\right\|_F^2$.

4 SHIFT

4.1 Overall Framework

As shown in Figure 2, SHIFT has four components operating in one loop. A base forecaster produces an initial prediction (see Appendix A); a generator expands it into M complete future hypotheses

$\{\boldsymbol{Z}_m^{(0)}\}$ with a prior belief $\boldsymbol{p}_0$; a causal filter updates the belief as blocks are revealed; and an assimilator updates the hypotheses themselves. At stage k the system commits the belief-weighted mixture on the current block,

$$\tilde{\boldsymbol{Y}}_{\mathcal{B}_k} = \sum_{m=1}^{M} \boldsymbol{p}_{k-1,m}\, \boldsymbol{Z}_m^{(k-1)}[\mathcal{B}_k], \tag{1}$$

then observes the truth of that block, producing per-hypothesis residuals that drive two updates: the belief moves from $\boldsymbol{p}_{k-1}$ to $\boldsymbol{p}_k$ (Section 4.3), and every hypothesis' unreleased portion moves from $\boldsymbol{Z}_m^{(k-1)}$ to $\boldsymbol{Z}_m^{(k)}$ (Section 4.4). Training adds a teacher that watches the full target and distills the posterior the evidence ultimately justifies into every intermediate belief (Section 4.5). All components are trained jointly by the objective of Section 4.6.

4.2 Hypothesis bank as low-rank residual fields

This subsection describes how history-conditioned hypotheses are generated around a shared point forecast. A base forecaster f_θ (a lightweight multiscale decomposition model which is shown in Appendix Section A) produces $\hat{\boldsymbol{Y}} = f_\theta(\boldsymbol{X})$. Each hypothesis adds to the base a smooth low-rank residual field, assembled from temporal shapes shared across variables and per-variable coefficients:

$$\boldsymbol{S}_m = \mathrm{normalize}_{\mathrm{rms}}\Big(g_{\mathrm{shape}}\big([\boldsymbol{p}_t^{\mathrm{pos}}; \boldsymbol{e}_m]\big)\Big) \in \mathbb{R}^{H\times r}, \qquad \boldsymbol{c}_{n,m} = \tanh\Big(g_{\mathrm{coef}}\big([\boldsymbol{h}_n; \boldsymbol{e}_m]\big)\Big) \in \mathbb{R}^{r}, \tag{2}$$

$$\boldsymbol{Z}_m^{(0)}[t,n] = \hat{\boldsymbol{Y}}[t,n] + \beta\,\sigma_n \sum_{j=1}^{r} \boldsymbol{S}_m[t,j]\,\boldsymbol{c}_{n,m,j}, \tag{3}$$

where $\boldsymbol{p}_t^{\mathrm{pos}}$ denotes sinusoidal positional features, $\boldsymbol{h}_n \in \mathbb{R}^5$ summarizes the historical statistics and dynamics of variable n (e.g., mean and most recent value), σ_n is its historical scale, and $\beta \in (0,1)$ is a learned global amplitude. Shared temporal shapes are modulated by variable-specific coefficients, while normalization and bounded scaling keep perturbations proportional to each variable's historical scale. The prior belief is computed from global summary statistics:

$$\boldsymbol{p}_0 = \mathrm{softmax}\Big(g_{\mathrm{prior}}\big([\bar{\boldsymbol{h}};\, \mu(\hat{\boldsymbol{Y}});\, \sigma(\hat{\boldsymbol{Y}})]\big)\Big), \tag{4}$$

4.3 Causal belief filtering

This subsection gives the update rule that turns a revealed block into a new belief. After committing stage k, the model observes $\boldsymbol{Y}_{\mathcal{B}_k}$ and forms per-hypothesis block residuals $\boldsymbol{R}_m = \boldsymbol{Y}_{\mathcal{B}_k} - \boldsymbol{Z}_m^{(k-1)}[\mathcal{B}_k]$. Each hypothesis receives an evidence score with two parts—a learned calibrator reading causal residual summaries, and a fixed likelihood energy:

$$\delta_{k,m} = \underbrace{g_{\mathrm{ev}}\big([\phi(\boldsymbol{R}_m); \boldsymbol{e}_m]\big)}_{\text{learned evidence}} - \underbrace{\mathcal{E}_{k,m}/\sigma_{\mathrm{obs}}}_{\text{likelihood energy}}, \qquad \mathcal{E}_{k,m} = \frac{1}{|\mathcal{B}_k|N}\big\|\boldsymbol{R}_m\big\|_F^2, \tag{5}$$

where $\phi(\cdot)$ collects six causal summaries (absolute error, squared error, bias, cross-variable consistency, roughness, deployment progress) and σ_{obs} is a learned positive observation scale. The belief is then updated log-linearly with a discount:

$$\ell_k = \gamma\,\ell_{k-1} + \delta_k/\tau_P, \qquad \boldsymbol{p}_k = \mathrm{softmax}(\ell_k), \qquad \ell_0 = \log \boldsymbol{p}_0. \tag{6}$$

The energy term serves as a Gaussian-likelihood anchor, encouraging the belief to favor hypotheses that better match the revealed observations. The learned calibrator complements this isotropic error signal by capturing richer residual patterns, such as directional bias and cross-variable structure. The discount factor γ further reduces the influence of stale evidence, allowing the filter to adapt to distribution changes within the forecast horizon.

4.4 Hypotheses as state: recurrent assimilation

This subsection updates the hypotheses themselves, which pure reweighting cannot do: a level shift observed in one block moves *every* hypothesis's unreleased trajectory. SHIFT maintains a latent

state per (hypothesis, variable) pair, folds each revealed residual into it with a GRU (Cho et al., 2014), and decodes a correction applied only to unreleased steps:

$$\begin{aligned} \boldsymbol{s}_{m,n} &\leftarrow \mathrm{GRU}\Big(\psi\big(\boldsymbol{R}_m, n\big),\ \boldsymbol{s}_{m,n}\Big), \\ \boldsymbol{Z}_m^{(k)}[t,n] &= \boldsymbol{Z}_m^{(k-1)}[t,n] + \sigma(a)\, g_{\mathrm{dec}}\big(\boldsymbol{s}_{m,n}, \boldsymbol{e}_m, t\big) \qquad \text{for } t > kS, \end{aligned} \tag{7}$$

where $\psi(\boldsymbol{R}_m, n)$ summarizes the block residual by four causal statistics (mean, last value, slope, dispersion) embedded with $\boldsymbol{e}_m$, $\sigma(a)$ is a learned global gate, and g_{dec} is zero-initialized so assimilation starts as the identity. Filtering (Eq. 6) decides *which* future to trust; assimilation (Eq. 7) decides *what each future now looks like*. Ablating either collapses SHIFT to a familiar baseline: assimilation alone is a single-trajectory corrector, filtering alone is fixed-particle reweighting.

4.5 Hindsight teachers

This subsection constructs the training-only supervision for the filter, whose correct output is unknowable causally. The teacher observes the full target and forms the posterior the evidence ultimately justifies—a soft argmin over hypotheses at temperature τ_T:

$$\boldsymbol{q}_m^\star = \mathrm{softmax}_m\Big(-\frac{1}{\tau_T}\,\mathrm{MSE}\left(\boldsymbol{Z}_m^{(0)}, \boldsymbol{Y}\right)\Big), \qquad \mathrm{MSE}\left(\boldsymbol{Z}_m^{(0)}, \boldsymbol{Y}\right) = \frac{1}{HN}\big\|\boldsymbol{Z}_m^{(0)} - \boldsymbol{Y}\big\|_F^2. \tag{8}$$

The isotropic energy makes the teacher's notion of the right hypothesis a mean-squared one: it discriminates amplitude-dominated mode differences well and timing-dominated ones less so—a bias we accept for its simplicity and state up front. Two distillation signals supervise the student:

$$\mathcal{L}_{\mathrm{bridge}} = \frac{1}{2(K-1)}\sum_{k=1}^{K-1}\mathbb{E}_b\,\mathrm{KL}(\mathrm{sg}(\boldsymbol{q}_b^\star)\,\|\,\boldsymbol{p}_{k,b}) + \frac{1}{2(K-1)}\sum_{k=1}^{K-1}\mathbb{E}_b\,\mathrm{KL}\big(\mathrm{sg}(\boldsymbol{q}_{k,b}^\star)\,\|\,\boldsymbol{p}_{k,b}\big)\,, \tag{9}$$

where $\mathbb{E}_b$ denotes the mini-batch average, sg stops gradients, and $\boldsymbol{q}_k^\star$ is a *stage oracle*: the hindsight posterior recomputed after the same assimilation step, restricted to the still-unseen future. The global teacher tells the student where the belief should end up; the stage oracle tells it what the best committable belief is *right now*, given what assimilation has done to the bank. Distillation applies only to post-evidence beliefs $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{K-1}$; the bridge loss is not applied directly to the history-only prior $\boldsymbol{p}_0$. Teacher distributions are stop-gradient targets in both distillation terms. Teacher calculations provide loss supervision or diagnostics; no teacher distribution enters the causal belief or assimilation updates. For $K = 1$, the bridge loss is zero because no feedback update is available.

4.6 Training objective

SHIFT is trained with exactly three loss terms: initial prediction, progressive suffix prediction, and the hindsight bridge of Eq. 9. Let B denote the mini-batch size. We write MSE for the mean squared error over the batch, time, and variable dimensions of its arguments. The initial prediction is the prior mixture $\boldsymbol{P}_0 = \sum_m p_{0,m}\boldsymbol{Z}_m^{(0)}$, supervised by a horizon-weighted absolute error,

$$\mathcal{L}_{\mathrm{base}}^{w}(\boldsymbol{P}_0, \boldsymbol{Y}) = \frac{1}{BHN}\sum_{b=1}^{B}\sum_{t=1}^{H}\sum_{n=1}^{N} w_t\,|(P_0)_{btn} - Y_{btn}|, \qquad w_t = t^{-1/2}.$$

The weights are applied before the element-wise mean, without normalization by their sum. After feedback update k, the updated belief and assimilated hypotheses define the prediction on the still-pending suffix. Its error is averaged across feedback stages:

$$\mathcal{L}_{\mathrm{prog}} = \frac{1}{K-1}\sum_{k=1}^{K-1}\mathrm{MSE}\left(\sum_m p_{k,m}\boldsymbol{Z}_m^{(k)}[>k], \boldsymbol{Y}_{>k}\right). \tag{10}$$

Here $[>k]$ denotes all steps after the k-th revealed block. Each stage first averages over its own remaining suffix and the batch; the $K-1$ stage means then receive equal weight. An empty stage average is defined as zero. Thus, our total training objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{base}}^{w}(\boldsymbol{P}_0, \boldsymbol{Y}) + \lambda_{\mathrm{prog}}\mathcal{L}_{\mathrm{prog}} + \lambda_{\mathrm{bridge}}\mathcal{L}_{\mathrm{bridge}}, \tag{11}$$

where λ_{prog} and λ_{bridge} are the weight factors. The base term trains the initial mixture; the progressive term trains the revised pending forecasts; the bridge trains post-feedback beliefs against the two detached teacher targets. These terms jointly optimize the backbone, hypothesis generator, evidence filter, and assimilator through their differentiable student paths. The initial and deployed MSE values recorded for monitoring are diagnostics, not additional training terms.

5 Experiments

5.1 Experiment Setup

Dataset. We evaluate our method on eleven real-world multivariate time-series datasets, including ETTh1, ETTh2, ETTm1, ETTm2, Weather, AQWan, CzeLan, ZafNoo, Electricity, AQShunyi, and Exchange. These datasets span a broad range of application domains, including energy, meteorology, air-quality monitoring, environmental sensing, and financial markets.

Baseline. We compare our method with two groups of representative baselines. For **online time-series forecasting and adaptation**, we include Proceed (Zhao & Shen, 2025), SOLID (Chen et al., 2024), FSNet (Pham et al., 2023), DynaME (Hong et al., 2026), and COSA (Im & Kwon, 2026), which are designed to adapt forecasting models to evolving temporal patterns, concept drift, or distribution shifts using newly observed data or test-time feedback. For **standard time-series forecasting**, we consider DistDF (Wang et al., 2026a), KUP-BI (Liu et al., 2026a), GFMixer (Zhang et al., 2026a), SEMixer (Zhang et al., 2026b), iTransformer (Liu et al., 2024), and xPatch (Stitsyuk & Choi, 2025). These methods are trained in an offline manner and directly generate future predictions from historical observations.

Comparison Setup. All experiments are conducted on NVIDIA GeForce RTX 4090 GPUs with 24 GB of memory. In the main experiments, we set the input window length to 336. The prediction horizons are set to 96, 192, 336, and 720. For the $H = 96$ setting, the feedback block length is set to $S = 24$; for $H \geq 192$, we set $S = 96$. We train all models using the Adam optimizer with a fixed learning rate of 2×10^{-4}, a batch size of 32, and a maximum of 30 epochs. Early stopping is applied based on validation-set performance. $\lambda_{\text{prog}} = 1$ and $\lambda_{\text{bridge}} = 0.2$.

5.2 Main Results

As shown in Table 1, the competing baselines exhibit noticeably different behaviors across datasets. Among the online forecasting and adaptation methods, Proceed and COSA are generally the most competitive: Proceed performs particularly well on several ETT benchmarks, while COSA shows clear advantages on Exchange, indicating that online adaptation can be effective when recent feedback provides informative signals about distribution changes. Nevertheless, their performance varies considerably across datasets, suggesting that the effectiveness of adaptation depends on the characteristics of the underlying distribution shift and the quality of available feedback. In comparison, standard forecasting methods provide more stable performance across the benchmark suite. In particular, GFMixer and SEMixer are consistently competitive on the ETT and Weather datasets, while xPatch performs strongly on Electricity and the air-quality datasets, highlighting the benefits of explicitly modeling temporal structures and local patterns. **Compared with the strongest baseline on each dataset, SHIFT reduces MSE and MAE by an average of approximately 25.2% and 15.1%, respectively**. The improvements are especially pronounced on Exchange, where the reductions reach **75.6%** in MSE and **48.5%** in MAE, as well as on Weather with reductions of 37.3% and 24.6%, and ETTm2 with reductions of 33.3% and 19.1%. These consistent gains across heterogeneous datasets demonstrate that **SHIFT** provides a more robust solution than either online adaptation or conventional forecasting alone.

5.3 Component Ablation Study

The ablation study examines the contribution of each component in **SHIFT**. **Static bank** keeps both candidate trajectories and their prior weights fixed, while **Filtering only** updates hypothesis weights according to revealed feedback without modifying the trajectories. **Assimilation only** instead revises the remaining trajectories while retaining the prior weights. **No teacher** combines filtering

Table 1: Forecasting performance comparison. The best results are highlighted in **red** and the second-best results in blue. In the header, denotes online forecasting models, and denotes standard forecasting models. "iTrans*" denotes iTransformer.

Models		SHIFT (Ours)		Proceed (2025)		SOLID (2024)		FSNet (2023)		DynaME (2026)		COSA (2026)		DistDF (2026)		KUP-BI (2026)		GFMixer (2026)		SEMixer (2026)		iTrans* (2024)		xPatch (2025)	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.348**	**0.380**	0.444	0.443	0.444	0.442	0.600	0.529	0.416	0.420	0.501	0.483	0.380	0.405	0.438	0.440	0.370	0.393	0.376	0.394	0.386	0.405	0.380	0.403
	192	**0.399**	**0.411**	0.496	0.476	0.501	0.477	0.700	0.591	0.481	0.451	0.568	0.529	0.416	0.425	0.467	0.461	0.413	0.418	0.418	0.419	0.441	0.436	0.425	0.430
	336	**0.411**	**0.415**	0.545	0.507	0.557	0.512	0.796	0.640	0.500	0.504	0.648	0.576	0.447	0.449	0.503	0.489	0.439	0.436	0.444	0.435	0.487	0.458	0.444	0.439
	720	**0.393**	**0.423**	0.658	0.579	0.671	0.586	1.289	0.806	0.554	0.556	0.721	0.623	0.475	0.484	0.542	0.529	0.470	0.475	0.444	0.456	0.503	0.491	0.534	0.509
	Avg	**0.388**	**0.407**	0.536	0.501	0.543	0.504	0.846	0.641	0.488	0.483	0.609	0.553	0.430	0.441	0.487	0.480	0.423	0.430	0.420	0.426	0.454	0.448	0.446	0.445
ETTh2	96	**0.182**	**0.274**	0.228	0.322	0.231	0.322	0.388	0.409	0.304	0.367	0.239	0.333	0.288	0.344	0.354	0.401	0.291	0.340	0.278	0.336	0.297	0.349	0.280	0.343
	192	**0.295**	**0.349**	0.300	0.359	0.305	0.356	0.484	0.470	0.406	0.429	0.299	0.375	0.350	0.384	0.377	0.411	0.353	0.380	0.343	0.374	0.380	0.400	0.345	0.386
	336	**0.282**	**0.344**	0.332	0.392	0.337	0.394	0.510	0.500	0.385	0.425	0.327	0.391	0.381	0.411	0.378	0.412	0.373	0.399	0.389	0.409	0.428	0.432	0.382	0.415
	720	**0.292**	**0.356**	0.452	0.463	0.442	0.456	0.617	0.562	0.414	0.459	0.436	0.453	0.417	0.475	0.442	0.452	0.397	0.428	0.398	0.431	0.427	0.445	0.408	0.443
	Avg	**0.263**	**0.331**	0.328	0.384	0.329	0.382	0.500	0.485	0.378	0.420	0.325	0.388	0.359	0.403	0.388	0.419	0.354	0.387	0.352	0.388	0.383	0.407	0.354	0.397
ETTm1	96	**0.211**	**0.287**	0.345	0.378	0.347	0.377	0.524	0.471	0.440	0.422	0.364	0.389	0.312	0.359	0.356	0.379	0.289	0.333	0.294	0.333	0.334	0.368	0.383	0.401
	192	**0.321**	**0.358**	0.403	0.412	0.407	0.413	0.736	0.548	0.562	0.459	0.454	0.443	0.349	0.383	0.371	0.390	0.332	0.360	0.333	0.360	0.377	0.391	0.431	0.427
	336	**0.335**	**0.367**	0.447	0.441	0.448	0.440	1.030	0.649	0.796	0.553	0.526	0.484	0.384	0.402	0.396	0.408	0.372	0.384	0.395	0.395	0.426	0.420	0.483	0.455
	720	**0.376**	**0.393**	0.506	0.477	0.509	0.475	1.168	0.710	1.392	0.684	0.610	0.538	0.435	0.432	0.511	0.459	0.421	0.413	0.421	0.414	0.491	0.459	0.486	0.474
	Avg	**0.311**	**0.351**	0.425	0.427	0.428	0.426	0.865	0.595	0.797	0.530	0.488	0.464	0.370	0.394	0.409	0.409	0.353	0.372	0.361	0.375	0.407	0.410	0.446	0.439
ETTm2	96	**0.098**	**0.199**	0.164	0.261	0.187	0.283	0.321	0.351	0.297	0.337	0.162	0.266	0.176	0.263	0.184	0.269	0.165	0.248	0.168	0.251	0.180	0.264	0.291	0.343
	192	**0.175**	**0.263**	0.203	0.290	0.256	0.329	0.521	0.426	0.559	0.441	0.206	0.301	0.241	0.312	0.243	0.310	0.225	0.287	0.221	0.288	0.250	0.309	0.372	0.392
	336	**0.173**	**0.264**	0.233	0.313	0.303	0.359	0.706	0.497	0.755	0.516	0.253	0.332	0.294	0.351	0.301	0.344	0.281	0.324	0.274	0.322	0.412	0.348	0.423	0.433
	720	**0.183**	**0.274**	0.348	0.387	0.426	0.424	0.921	0.577	0.773	0.562	0.349	0.387	0.392	0.403	0.384	0.396	0.361	0.378	0.362	0.376	0.311	0.407	0.430	0.451
	Avg	**0.158**	**0.250**	0.237	0.313	0.293	0.349	0.617	0.463	0.596	0.464	0.243	0.321	0.275	0.332	0.278	0.330	0.258	0.309	0.256	0.309	0.288	0.332	0.409	0.405
Weather	96	**0.094**	**0.128**	0.155	0.205	0.156	0.206	0.294	0.297	0.229	0.279	0.177	0.230	0.185	0.242	0.166	0.215	0.146	0.191	0.147	0.184	0.174	0.214	0.178	0.221
	192	**0.160**	**0.211**	0.197	0.243	0.199	0.244	0.597	0.418	0.353	0.371	0.231	0.273	0.253	0.271	0.244	0.282	0.198	0.239	0.193	0.230	0.221	0.254	0.231	0.263
	336	**0.152**	**0.203**	0.252	0.284	0.253	0.285	0.823	0.509	0.393	0.401	0.301	0.316	0.297	0.313	0.298	0.317	0.258	0.291	0.243	0.270	0.278	0.296	0.283	0.300
	720	**0.160**	**0.217**	0.319	0.333	0.320	0.334	1.021	0.593	0.473	0.441	0.376	0.362	0.394	0.391	0.373	0.360	0.325	0.343	0.318	0.324	0.358	0.347	0.360	0.350
	Avg	**0.141**	**0.190**	0.231	0.267	0.232	0.267	0.684	0.454	0.362	0.373	0.271	0.295	0.282	0.304	0.270	0.294	0.232	0.266	0.225	0.252	0.258	0.278	0.264	0.284
AQWan	96	**0.561**	**0.383**	0.783	0.495	0.884	0.540	1.244	0.598	0.935	0.542	0.751	0.490	0.730	0.469	0.766	0.478	0.733	0.462	0.719	0.455	0.779	0.494	0.717	0.469
	192	**0.718**	**0.467**	0.872	0.525	0.953	0.557	1.395	0.621	0.969	0.568	0.860	0.524	0.789	0.501	0.825	0.497	0.765	0.476	0.771	0.478	0.817	0.508	0.763	0.490
	336	**0.719**	**0.466**	0.909	0.538	0.944	0.560	1.119	0.596	1.407	0.623	0.867	0.534	0.788	0.506	0.801	0.485	0.789	0.489	0.803	0.491	0.812	0.497	0.789	0.505
	720	**0.755**	**0.478**	0.972	0.567	1.007	0.587	1.096	0.584	1.346	0.663	1.040	0.590	0.862	0.534	0.840	0.497	0.868	0.518	0.873	0.518	0.886	0.537	0.861	0.532
	Avg	**0.688**	**0.449**	0.884	0.531	0.947	0.561	1.213	0.600	1.164	0.599	0.880	0.534	0.792	0.502	0.808	0.489	0.789	0.486	0.792	0.485	0.824	0.509	0.783	0.499
CzeLan	96	**0.106**	**0.166**	0.244	0.279	0.240	0.280	0.322	0.355	0.297	0.320	0.185	0.236	0.178	0.233	0.229	0.262	0.181	0.228	0.181	0.216	0.174	0.215	0.173	0.228
	192	**0.176**	**0.234**	0.286	0.321	0.318	0.343	0.388	0.399	0.422	0.401	0.223	0.264	0.225	0.263	0.243	0.280	0.209	0.238	0.212	0.241	0.240	0.261	0.204	0.256
	336	**0.175**	**0.237**	0.355	0.374	0.384	0.392	0.628	0.523	0.484	0.447	0.266	0.301	0.250	0.281	0.258	0.288	0.236	0.266	0.248	0.268	0.268	0.281	0.240	0.290
	720	**0.203**	**0.263**	0.417	0.421	0.367	0.396	0.632	0.493	0.428	0.444	0.339	0.352	0.290	0.331	0.405	0.385	0.273	0.307	0.301	0.306	0.276	0.298	0.268	0.315
	Avg	**0.165**	**0.225**	0.325	0.349	0.327	0.353	0.493	0.443	0.408	0.403	0.253	0.288	0.236	0.277	0.284	0.304	0.225	0.260	0.235	0.258	0.240	0.264	0.222	0.272
ZafNoo	96	**0.297**	**0.315**	0.501	0.443	0.648	0.506	0.628	0.508	0.602	0.496	0.451	0.416	0.453	0.404	0.495	0.417	0.444	0.398	0.450	0.403	0.490	0.422	0.439	0.414
	192	**0.438**	**0.409**	0.613	0.498	0.680	0.529	0.791	0.578	0.642	0.531	0.571	0.464	0.523	0.445	0.531	0.440	0.509	0.439	0.515	0.443	0.552	0.463	0.500	0.449
	336	**0.426**	**0.403**	0.657	0.525	0.728	0.549	0.968	0.635	0.619	0.531	0.676	0.502	0.568	0.476	0.599	0.491	0.561	0.469	0.566	0.473	0.596	0.486	0.562	0.481
	720	**0.441**	**0.418**	0.695	0.543	0.697	0.538	1.084	0.659	0.704	0.561	0.672	0.527	0.620	0.500	0.628	0.487	0.618	0.503	0.617	0.501	0.659	0.524	0.618	0.519
	Avg	**0.400**	**0.386**	0.616	0.502	0.688	0.530	0.868	0.595	0.642	0.530	0.593	0.477	0.541	0.456	0.563	0.459	0.533	0.452	0.537	0.455	0.574	0.474	0.530	0.466
Electricity	96	**0.107**	**0.207**	0.130	0.224	0.130	0.224	0.184	0.292	0.149	0.243	0.156	0.256	0.137	0.238	0.153	0.251	0.139	0.236	0.139	0.233	0.136	0.232	0.130	0.226
	192	**0.139**	**0.236**	0.178	0.275	0.180	0.279	0.224	0.326	0.180	0.277	0.171	0.272	0.158	0.249	0.167	0.259	0.157	0.257	0.154	0.246	0.154	0.247	0.150	0.243
	336	**0.142**	**0.243**	0.212	0.315	0.195	0.293	0.241	0.342	0.200	0.296	0.188	0.289	0.169	0.274	0.178	0.274	0.172	0.265	0.171	0.262	0.170	0.263	0.169	0.265
	720	**0.151**	**0.257**	0.232	0.341	0.244	0.345	0.273	0.363	0.219	0.312	0.233	0.326	0.212	0.300	0.232	0.306	0.198	0.293	0.209	0.294	0.210	0.297	0.206	0.299
	Avg	**0.135**	**0.236**	0.188	0.289	0.187	0.285	0.230	0.331	0.187	0.282	0.187	0.286	0.169	0.265	0.183	0.273	0.166	0.263	0.168	0.259	0.168	0.260	0.164	0.258
AQShunyi	96	**0.476**	**0.395**	0.701	0.508	0.770	0.541	1.024	0.595	0.825	0.550	0.657	0.496	0.651	0.472	0.679	0.497	0.649	0.473	0.636	0.467	0.670	0.484	0.634	0.479
	192	**0.628**	**0.478**	0.806	0.539	0.856	0.566	1.009	0.600	0.912	0.579	0.704	0.519	0.697	0.516	0.723	0.508	0.683	0.491	0.682	0.487	0.701	0.497	0.666	0.495
	336	**0.627**	**0.479**	0.800	0.547	0.845	0.573	0.892	0.594	0.985	0.627	0.737	0.536	0.712	0.513	0.718	0.488	0.697	0.500	0.711	0.501	0.714	0.507	0.686	0.506
	720	**0.655**	**0.487**	0.851	0.567	0.855	0.580	0.880	0.573	1.074	0.652	0.882	0.582	0.758	0.525	0.734	0.506	0.773	0.541	0.760	0.525	0.766	0.527	0.733	0.527
	Avg	**0.596**	**0.460**	0.790	0.540	0.831	0.565	0.951	0.591	0.949	0.602	0.745	0.533	0.705	0.507	0.714	0.500	0.701	0.501	0.698	0.495	0.712	0.504	0.680	0.502
Exchange	96	**0.025**	**0.108**	0.138	0.264	0.151	0.274	0.150	0.266	0.333	0.406	0.080	0.196	0.089	0.212	0.231	0.350	0.084	0.202	0.088	0.207	0.086	0.206	0.093	0.215
	192	**0.103**	**0.230**	0.274	0.375	0.255	0.361	0.331	0.412	0.435	0.505	0.142	0.272	0.188	0.313	0.313	0.422	0.193	0.311	0.183	0.304	0.177	0.299	0.223	0.344
	336	**0.091**	**0.215**	0.367	0.439	0.402	0.458	0.616	0.575	0.418	0.501	0.238	0.353	0.341	0.427	0.542	0.550	0.356	0.429	0.328	0.413	0.331	0.417	0.366	0.445
	720	**0.102**	**0.227**	0.888	0.710	0.932	0.712	1.452	0.887	0.896	0.709	0.853	0.694	1.199	0.827	1.807	1.009	1.422	0.846	1.067	0.821	0.847	0.691	1.242	0.831
	Avg	**0.080**	**0.195**	0.417	0.447	0.435	0.451	0.637	0.535	0.521	0.530	0.328	0.379	0.454	0.444	0.723	0.583	0.514	0.447	0.200	0.308	0.360	0.403	0.481	0.459

and assimilation without hindsight-teacher supervision, and **Single corrector** performs assimilation using only one candidate trajectory. **Full SHIFT** integrates filtering, assimilation, and hindsight-teacher supervision, where the teachers are used only during training.

As shown in Table 2, the static bank performs consistently worse, indicating that fixed hypotheses cannot effectively exploit newly revealed observations. Filtering alone provides moderate improvements, but remains substantially weaker than assimilation across all four datasets, showing that reweighting existing candidates is less effective than directly revising their future trajectories. Combining filtering and assimilation further improves performance, while adding hindsight-teacher supervision yields the strongest overall results: Full SHIFT achieves the best performance on seven of the eight reported metrics. This confirms that assimilation is the primary source of feedback adaptation, whereas filtering and teacher supervision provide complementary gains.

The accompanying teacher-ablation results in Figure 3 further isolate the effect of training supervision. Using either the global or stage-wise teacher alone already improves over the no-teacher variant on most metrics, while combining both produces the most consistent performance across

Table 2: Component ablation of SHIFT under the PFF protocol. All runs use $L = 336$, $H = 96$, and feedback blocks of length 24. M is the number of hypotheses.

Variant	M	F	A	T	ETTh1		ETTm2		Weather		CzeLan	
					MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Static bank	8	✗	✗	✗	0.3742	0.3939	0.1714	0.2591	0.1571	0.2074	0.1724	0.2350
Filtering only	8	✓	✗	✗	0.3712	0.3982	0.1432	0.2444	0.1417	0.1938	0.1561	0.2240
Assimilation only	8	✗	✓	✗	0.3487	0.3834	0.1015	0.2007	0.0970	0.1341	0.1079	0.1734
No teacher	8	✓	✓	✗	0.3515	0.3832	0.1011	0.2028	0.0963	0.1314	0.1088	0.1722
Single corrector	1	✗	✓	✗	0.3515	0.3816	0.0991	**0.1983**	0.0969	0.1309	0.1081	0.1701
Full SHIFT	8	✓	✓	✓	**0.3469**	**0.3794**	**0.0985**	0.1987	**0.0955**	**0.1294**	**0.1064**	**0.1672**

F/A/T denote filtering, assimilation, and hindsight-teacher supervision, respectively. Filtering off retains the learned prior $p_k = p_0$; assimilation off retains $Z_k = Z_0$. The single corrector maintains exactly one trajectory. Teachers supervise training only.

ETTh1, ETTm2, Weather, and CzeLan. This suggests that the two teacher signals provide complementary supervision for learning feedback-driven corrections, rather than relying on a single form of hindsight guidance.

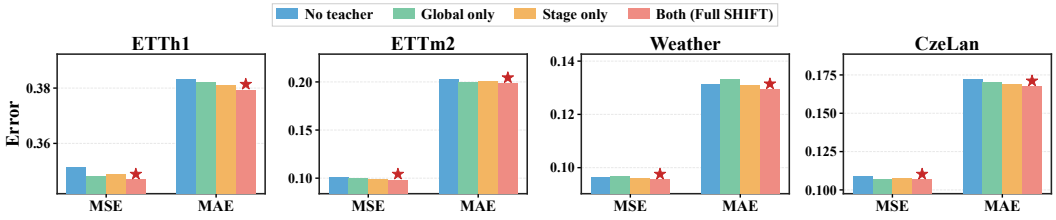

Figure 3: Ablation study of hindsight-teacher supervision across different datasets.

5.4 Stage-wise Mechanism Analysis

We examine how accumulated feedback improves predictions of the same fixed target and which update mechanism drives the gain. Specifically, the target is fixed to the final block $B_4 = [72, 96)$, while Stage k denotes the number of revealed feedback blocks. As shown in Figure 4, **Full SHIFT** progressively reduces prediction error as more feedback becomes available, with MSE decreasing by 9.1%, 55.0%, and 49.4% from Stage 0 to Stage 3 on ETTh1, Weather, and CzeLan, respectively. At Stage 3, it achieves 10.0%, 44.9%, and 37.0% lower MSE than **Filtering only**. The accompanying update decomposition in Table 3 further shows that most of the improvement comes from assimilation, while filtering contributes only marginal direct gains. Together, these results indicate that effectively revising candidate trajectories, rather than merely reweighting them, is the primary mechanism for exploiting newly observed feedback.

5.5 Hyperparameter Experiments

Effect of Feedback Block Size We study the effect of feedback granularity by varying the feedback block size $S \in \{12, 24, 48, 96\}$. A smaller S provides more frequent observations and therefore allows **SHIFT** to revise its forecasts more often, while a larger S reduces the number of feedback updates and lowers the associated computational overhead. As shown in Figure 5, smaller feedback blocks consistently yield better prediction accuracy across ETTh1, Weather, and CzeLan. In particular, $S = 12$ achieves the lowest deployed MSE on all three datasets, confirming that more frequent feedback enables finer-grained correction of the remaining trajectory. This improvement, however, comes with increased inference cost. The median inference latency decreases substantially as S grows, while the prediction error generally increases, revealing a clear accuracy–efficiency trade-off. The same tendency is also reflected in peak inference memory, which decreases with larger feedback blocks. We therefore adopt $S = 24$ as the default setting: it retains much of the benefit of frequent feedback while substantially reducing the update overhead compared with $S = 12$.

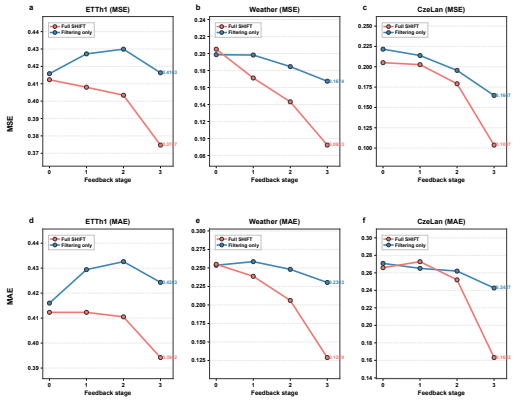

Figure 4: Fixed-target prediction error across successive feedback stages.

Table 3: Feedback-update decomposition. Δ_{F}, Δ_{A}, and Δ_{FA} denote MSE reductions from filtering, assimilation, and the full update.

Dataset	Target	Δ_{F}	Δ_{A}	Δ_{FA}
ETTh1	B_2	6.296e-06	2.146e-02	2.145e-02
	B_3	2.616e-06	2.289e-02	2.289e-02
	B_4	2.669e-06	2.869e-02	2.869e-02
Weather	B_2	9.798e-04	5.913e-02	5.912e-02
	B_3	8.675e-04	5.210e-02	5.209e-02
	B_4	6.263e-04	5.084e-02	5.088e-02
CzeLan	B_2	5.818e-07	7.280e-02	7.280e-02
	B_3	1.268e-06	7.659e-02	7.659e-02
	B_4	1.662e-06	7.534e-02	7.534e-02

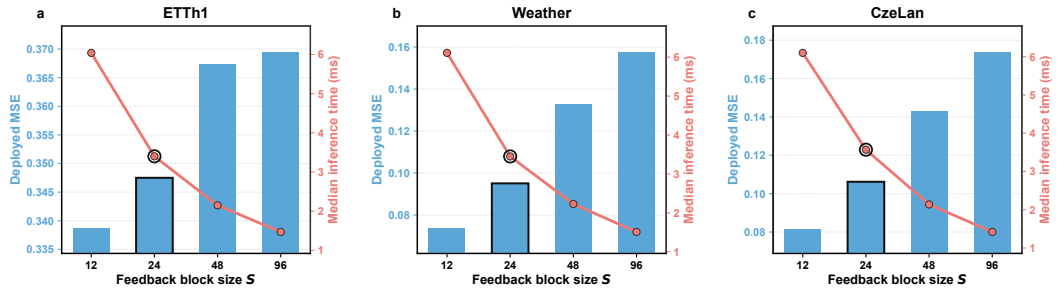

Figure 5: Experiment on sensitivity to feedback block size.

Effect of the Number of Hypotheses We vary the hypothesis-bank size $M \in \{1, 4, 8, 16, 32, 64, 128\}$ to study the trade-off between predictive diversity and computational cost. As shown in Figure 6, using multiple hypotheses generally improves forecasting performance over a single trajectory, while further enlarging the bank yields only limited and inconsistent gains. In contrast, inference latency increases noticeably for large M.

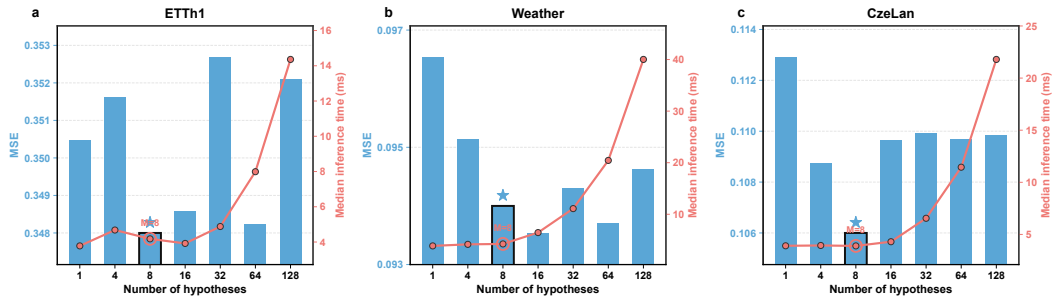

Figure 6: Experiment on sensitivity to the number of hypotheses.

6 Conclusion

We presented **SHIFT**, a progressive feedback forecasting framework that maintains multiple future hypotheses and progressively revises them as new feedback becomes available. **SHIFT** combines belief filtering, trajectory assimilation, and hindsight-teacher supervision to exploit partially revealed observations while preserving causal inference. Experiments across diverse time-series benchmarks demonstrate consistent improvements over conventional forecasting and alternative feedback-revision strategies, while ablation and mechanism analyses highlight the importance of trajectory assimilation and multi-hypothesis modeling.

REFERENCES

Apratim Bhattacharyya, Bernt Schiele, and Mario Fritz. Accurate and diverse sampling of sequences based on a "best of many" sample objective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8485–8493, 2018.

Christopher M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Aston University, 1994.

Fanpu Cao, Lu Dai, Jindong Han, and Hui Xiong. Enhancing multivariate time series forecasting with global temporal retrieval. In *International Conference on Learning Representations*, 2026.

Mouxiang Chen, Lefei Shen, Han Fu, Zhuo Li, Jianling Sun, and Chenghao Liu. Calibration of time-series forecasting: Detecting and adapting context-driven distribution shift. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 341–352. ACM, 2024. doi: 10.1145/3637528.3671926.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Seungha Hong, Sukang Chae, Suyeon Kim, Sanghwan Jang, and Hwanjo Yu. Dynamic multi-period experts for online time series forecasting. In *Proceedings of the ACM Web Conference 2026*, pp. 7669–7678. Association for Computing Machinery, 2026. doi: 10.1145/3774904.3792731.

Ao Hu, Liangjian Wen, Jiang Duan, Yong Dai, He Yan, Dongkai Wang, Jun Wang, Yukun Zhang, Ruoxi Jiang, and Zenglin Xu. PMDFormer: Patch-mean decoupling information transformer for long-term forecasting. In *International Conference on Learning Representations*, 2026.

Yuxuan Hu, Peiyuan Liu, Peng Zhu, Daizong Cheng, and Tao Dai. Adaptive multi-scale decomposition framework for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 17359–17367, 2025.

Songtao Huang, Zhen Zhao, Can Li, and Lei Bai. TimeKAN: KAN-based frequency decomposition learning architecture for long-term time series forecasting. In *International Conference on Learning Representations*, 2025.

Xiannan Huang, Shuhan Qiu, Jiayuan Du, and Chao Yang. Online time series prediction using feature adjustment. In *International Conference on Learning Representations*, 2026.

Jeonghwan Im and Hyuk-Yoon Kwon. COSA: Context-aware output-space adapter for test-time adaptation in time series forecasting. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=L7Z5wBMPrW.

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

Ying-yee Ava Lau, Zhiwen Shao, and Dit-Yan Yeung. Fast and slow streams for online time series forecasting without information leakage. In *International Conference on Learning Representations*, 2025.

Chong Liu, Yingjie Zhou, Hao Li, Pengyang Wang, Qingsong Wen, and Ce Zhu. Beyond extrapolation: Knowledge utilization paradigm with bidirectional inspiration for time series forecasting. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026a.

Kun Liu, Renjun Jia, Ruifeng Yang, Xirui Zeng, Yuqi Liang, and Cen Chen. Are global dependencies necessary? scalable time series forecasting via local cross-variate modeling. In *International Conference on Learning Representations*, 2026b.

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*, 2024.

Ziqi Liu. FreqMoE: Enhancing time series forecasting through frequency decomposition mixture of experts. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258, pp. 3430–3438, 2025.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. In *International Conference on Learning Representations*, 2016.

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.

Yiming Niu, Jinliang Deng, and Yongxin Tong. PhaseFormer: From patches to phases for efficient and effective time series forecasting. In *International Conference on Learning Representations*, 2026.

Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven C. H. Hoi. Learning fast and slow for online time series forecasting. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=q-PbpHD3EOk.

Artyom Stitsyuk and Jaesik Choi. xPatch: Dual-stream time series forecasting with exponential seasonal-trend decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20601–20609, 2025. doi: 10.1609/aaai.v39i19.34270.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557, 2009.

Hao Wang, Licheng Pan, Yuan Lu, Zhixuan Chu, Xiaoxi Li, Shuting He, Zhichao Chen, Haoxuan Li, Qingsong Wen, and Zhouchen Lin. DistDF: Time-series forecasting needs joint-distribution wasserstein alignment. In *International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=VrdLwUmzBy.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Chen Wang, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026b. doi: 10.1109/TPAMI.2026.3690845.

Qingsong Wen, Weiqi Chen, Liang Sun, Zhang Zhang, Liang Wang, Rong Jin, Tieniu Tan, et al. Onenet: Enhancing time series forecasting models under concept drift by online ensembling. *Advances in Neural Information Processing Systems*, 36:69949–69980, 2023.

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*, 2023.

Lin Zhang, Qing Li, and Jingmei Zhao. GFMixer: Decoupled temporal gradient and fourier-aware attention for time series forecasting. In *Proceedings of the ACM Web Conference 2026*, pp. 7156–7166. Association for Computing Machinery, 2026a. doi: 10.1145/3774904.3792345.

Xu Zhang, Qitong Wang, Peng Wang, and Wei Wang. SEMixer: Semantics enhanced mlp-mixer for multiscale mixing and long-term time series forecasting. *arXiv preprint arXiv:2602.16220*, 2026b.

Yi-Fan Zhang, Qingsong Wen, Xue Wang, Weiqi Chen, Liang Sun, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. OneNet: Enhancing time series forecasting models under concept drift by online ensembling. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Lifan Zhao and Yanyan Shen. Proactive model adaptation against concept drift for online time series forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2020–2031. ACM, 2025. doi: 10.1145/3690624.3709210.

A Lightweight Multiscale Decomposition Backbone

Given a multivariate history $\boldsymbol{X} \in \mathbb{R}^{L\times N}$, the backbone f_θ first separates it into a smooth trend and a residual component:

$$\boldsymbol{T} = \mathrm{MA}_{25}(\boldsymbol{X}), \qquad \boldsymbol{S} = \boldsymbol{X} - \boldsymbol{T}, \tag{12}$$

where MA_{25} denotes moving-average decomposition with a window size of 25. The backbone constructs six complementary forecasting branches: one linear trend projection, four residual branches with temporal filter sizes $\{3, 7, 15, 25\}$, and one periodic branch obtained by repeating the most recent cycle. We denote their predictions by $\{\boldsymbol{B}^{(j)}\}_{j=1}^{6}$.

The contribution of each branch is determined dynamically for every variable. Specifically, we construct $\boldsymbol{g}_n$ (such as mean and standard deviation) from compact statistical and temporal descriptors of each historical variable. A lightweight gating network produces the branch weights:

$$\boldsymbol{U}_{:,n} = \sum_{j=1}^{6} \left[\mathrm{Softmax}\big(g_{\mathrm{branch}}(\boldsymbol{g}_n)\big)\right]_j \boldsymbol{B}^{(j)}_{:,n}. \tag{13}$$

To capture cross-variable dependencies efficiently, we use a dynamic low-rank factor mixer. Let $a_{n,r}$ and $\ell_{n,r}$ denote the factor score and loading of variable n for latent factor r. The cross-variable correction is computed as

$$F_{h,r} = \sum_{n=1}^{N} a_{n,r} U_{h,n}, \qquad \Delta_{h,n} = \frac{1}{\sqrt{R}} \sum_{r=1}^{R} F_{h,r}\ell_{n,r}, \tag{14}$$

where R is the factor rank. This operation models variable interactions with complexity linear in the number of variables. Finally, a gated refinement network combines the mixed forecast, the periodic branch $\boldsymbol{B}^{\mathrm{cyc}}$, and their difference:

$$\begin{aligned} \overline{\boldsymbol{Y}} &= \boldsymbol{U} + \eta_1\, \boldsymbol{G}_{\mathrm{ch}} \odot \boldsymbol{\Delta}, \\ \hat{\boldsymbol{Y}} = f_\theta(\boldsymbol{X}) &= \overline{\boldsymbol{Y}} + \eta_2\, g_{\mathrm{ref}}\left([\overline{\boldsymbol{Y}}, \boldsymbol{B}^{\mathrm{cyc}}, \overline{\boldsymbol{Y}} - \boldsymbol{B}^{\mathrm{cyc}}]\right). \end{aligned} \tag{15}$$

The resulting $\hat{\boldsymbol{Y}} \in \mathbb{R}^{H\times N}$ serves as the initial forecast. **SHIFT** subsequently constructs multiple complete future hypotheses by adding structured low-rank residual fields around this forecast.

B Computational Complexity Analysis

We analyze the time and memory cost of **SHIFT** for a single forecasting episode; all costs scale linearly with the batch size. We use L for the input length, H for the horizon, N for the number of variables, M for the number of hypotheses, S for the feedback block length with $K = \lceil H/S \rceil$ stages, r for the rank of the residual fields, R for the factor rank of the backbone mixer, and d for the hidden width of the small networks (g_{shape}, g_{coef}, g_{ev}, g_{dec}) and of the GRU state $\boldsymbol{s}_{m,n}$.

One-time cost. The backbone is evaluated only once per episode. The moving-average decomposition, the branch projections, and the low-rank factor mixer cost $\mathcal{O}(LHN+HNR)$. The hypothesis generator evaluates g_{shape} for each (step, hypothesis) pair and g_{coef} for each (variable, hypothesis) pair, and assembles the residual fields in Equation (3), giving $\mathcal{O}\big(M(H+N)d^2 + MHNr\big)$. The prior in Equation (4) requires only $\mathcal{O}(HN)$ summary statistics and a small MLP.

Per-stage cost. At stage k, the causal filter computes the block residuals $\boldsymbol{R}_m$ and the summaries $\phi(\boldsymbol{R}_m)$ in $\mathcal{O}(MSN)$, evaluates g_{ev} in $\mathcal{O}(Md^2)$, and updates the belief in Equation (6) in $\mathcal{O}(M)$. The recurrent assimilator computes $\psi(\boldsymbol{R}_m, n)$ in $\mathcal{O}(MSN)$, updates the MN GRU states in $\mathcal{O}(MNd^2)$, and decodes corrections for the $H - kS$ unreleased steps in $\mathcal{O}\big(MN(H-kS)d\big)$. Committing the mixture in Equation (1) costs $\mathcal{O}(MSN)$.

Total inference cost. Summing over the K stages and using $KS \approx H$, the total inference time is

$$\mathcal{O}\Big(\underbrace{LHN + HNR}_{\text{backbone}} + \underbrace{M(H+N)d^2 + MHNr}_{\text{generator}} + \underbrace{MHN + KMNd^2 + KMNHd}_{\text{filter and assimilator}} \Big), \quad (16)$$

which is linear in L, H, N, and M. The inference memory is dominated by the hypothesis bank and the assimilation states, i.e., $\mathcal{O}\big(MN(H+d)\big)$, since the belief vector and the stage summaries require only $\mathcal{O}(M)$ and $\mathcal{O}(MN)$ storage. The number of learnable parameters is independent of N, and depends on M only through the hypothesis embeddings $\boldsymbol{e}_m$, because the GRU, the calibrator, and the decoder are shared across all (hypothesis, variable) pairs.

Training overhead. During training, the hindsight teachers add $\mathcal{O}(MHN)$ for the global teacher in Equation (8) and $\mathcal{O}(KMHN)$ for the $K-1$ stage oracles, while the two KL terms in Equation (9) add only $\mathcal{O}(KM)$. Because gradients are propagated through all feedback stages, the activation memory grows to $\mathcal{O}\big(KMN(H+d)\big)$. The teachers are used only for training and add no cost at inference.

Comparison with alternative feedback strategies. The per-stage update of **SHIFT** does not depend on the input length L, since the history is encoded only once and each feedback block is folded into compact latent states. In contrast, rolling re-forecasting re-runs the forecaster on an updated input window at every stage, costing $\mathcal{O}\big(K(LHN + HNR)\big)$ for the same backbone, and online adaptation methods perform additional forward and backward passes to update model parameters when feedback arrives. **SHIFT** performs no gradient computation or parameter update at inference.

Practical remarks. The K stages are inherently sequential because each update depends on the block revealed at that stage. In deployment, however, these updates are distributed over time as feedback arrives, so the latency relevant to each decision is a single per-stage update rather than the full episode. Within each stage, the $M \times N$ (hypothesis, variable) pairs are processed in parallel, so on GPUs the wall-clock time grows sublinearly in M and N until the hardware is saturated. This is consistent with the empirical trends in Figures 5 and 6: a smaller block length S increases K and therefore the latency, while the latency increases noticeably only when M becomes large.

C Experiment

C.1 Details of Baseline Methods

Online Forecasting and Adaptation Methods. These methods are designed for non-stationary forecasting scenarios, where temporal patterns or data distributions may evolve over time. Instead of keeping the forecasting model completely fixed during inference, they exploit newly observed samples, delayed feedback, or recent contextual information to adapt the forecasting process to distribution shifts and concept drift. The selected baselines cover several representative adaptation strategies, including parameter adaptation, contextual sample retrieval, continual learning, expert-based modeling, and output-space correction.

Proceed (Zhao & Shen, 2025) is a proactive adaptation framework designed for online time-series forecasting under concept drift. Rather than directly updating the forecasting model using potentially outdated feedback, Proceed estimates the concept drift between recently observed training samples and the current test sample and translates the estimated drift into model-parameter adjustments through an adaptation generator.

SOLID (Chen et al., 2024) is a sample-level contextualized adaptation method developed to mitigate context-driven distribution shifts. For each test sample, it retrieves contextually similar historical samples and performs lightweight adaptation of the prediction layer, allowing the forecasting model to better match the local test-time distribution.

FSNet (Pham et al., 2023) is an online forecasting framework inspired by fast-and-slow learning mechanisms. It combines rapid adaptation to recently observed patterns with the ability to retain and reuse previously acquired knowledge, enabling the forecasting model to respond to both newly emerging and recurring temporal patterns.

DynaME (Hong et al., 2026) addresses online forecasting through dynamic multi-period experts. It distinguishes between recurring and emerging concept shifts and dynamically exploits specialized experts associated with relevant historical temporal patterns, while maintaining the ability to accommodate previously unseen changes.

COSA (Im & Kwon, 2026) is a lightweight test-time adaptation approach that directly operates in the output space of a frozen forecasting model. It exploits recently observed information as contextual feedback and applies a gated residual correction to the original forecast, enabling efficient adaptation without extensively modifying the underlying forecasting backbone.

Standard Time-Series Forecasting Methods. These methods follow the conventional offline forecasting paradigm, where models are trained on a fixed training set and directly generate future predictions from historical observations at inference time. Unlike online adaptation methods, they do not rely on newly observed labels or perform continual model updates during inference. The selected baselines represent recent advances in distribution modeling, knowledge utilization, temporal and frequency representation learning, multiscale modeling, cross-variable dependency modeling, and decomposition-based forecasting.

DistDF (Wang et al., 2026a) improves direct forecasting by explicitly reducing the discrepancy between forecast and target distributions. In particular, it introduces a joint-distribution Wasserstein alignment objective to better capture dependencies within multi-step forecasting targets and can be integrated into existing forecasting architectures.

KUP-BI (Liu et al., 2026a) introduces a knowledge-utilization paradigm that augments conventional history-to-future forecasting with continuation-style information extracted exclusively from training trajectories. It constructs a historical knowledge library, retrieves an approximate continuation signal for the current input, and fuses this auxiliary information with the original forecasting representation.

GFMixer (Zhang et al., 2026a) is a dual-path forecasting architecture that jointly models temporal variations and frequency-domain structures. It combines temporal-gradient modeling with Fourier-aware representation learning to capture complementary temporal and spectral patterns for long-term forecasting.

SEMixer (Zhang et al., 2026b) is a lightweight multiscale forecasting architecture based on MLP-Mixer. It models temporal dependencies across multiple scales and progressively integrates information from different temporal resolutions, enabling effective representation of complex long-term dynamics.

iTransformer (Liu et al., 2024) revisits the Transformer architecture by treating individual variables rather than time steps as tokens. Self-attention is therefore used to explicitly model cross-variable dependencies, while temporal representations are learned within each variate token for multivariate time-series forecasting.

xPatch (Stitsyuk & Choi, 2025) is a dual-stream forecasting architecture based on seasonal–trend decomposition. It combines linear and nonlinear modeling streams with patch-based temporal representations to capture complementary temporal structures and improve long-term forecasting performance.

C.2 Relative Performance Across Forecasting Horizons

Under the $L = 96$ setting, we further conduct experiments with prediction horizons $H \in \{96, 192, 336, 720\}$ across five datasets. For a unified comparison, the MSE and MAE results are normalized into relative performance scores, with the best performance in each setting serving as the reference score of 100.

As shown in Figure 7, **SHIFT** consistently delivers strong and competitive performance across all five datasets and four forecasting horizons. Beyond this overall consistency, substantial improvements are observed in several challenging settings. At horizon 96, **SHIFT** reduces MSE by **31.6%** on ETTm2 and **38.7%** on Weather compared with the strongest competing results, together with corresponding MAE reductions of **17.5%** and **34.3%**. On ETTm1, it further achieves a **22.3%** reduction in MSE.

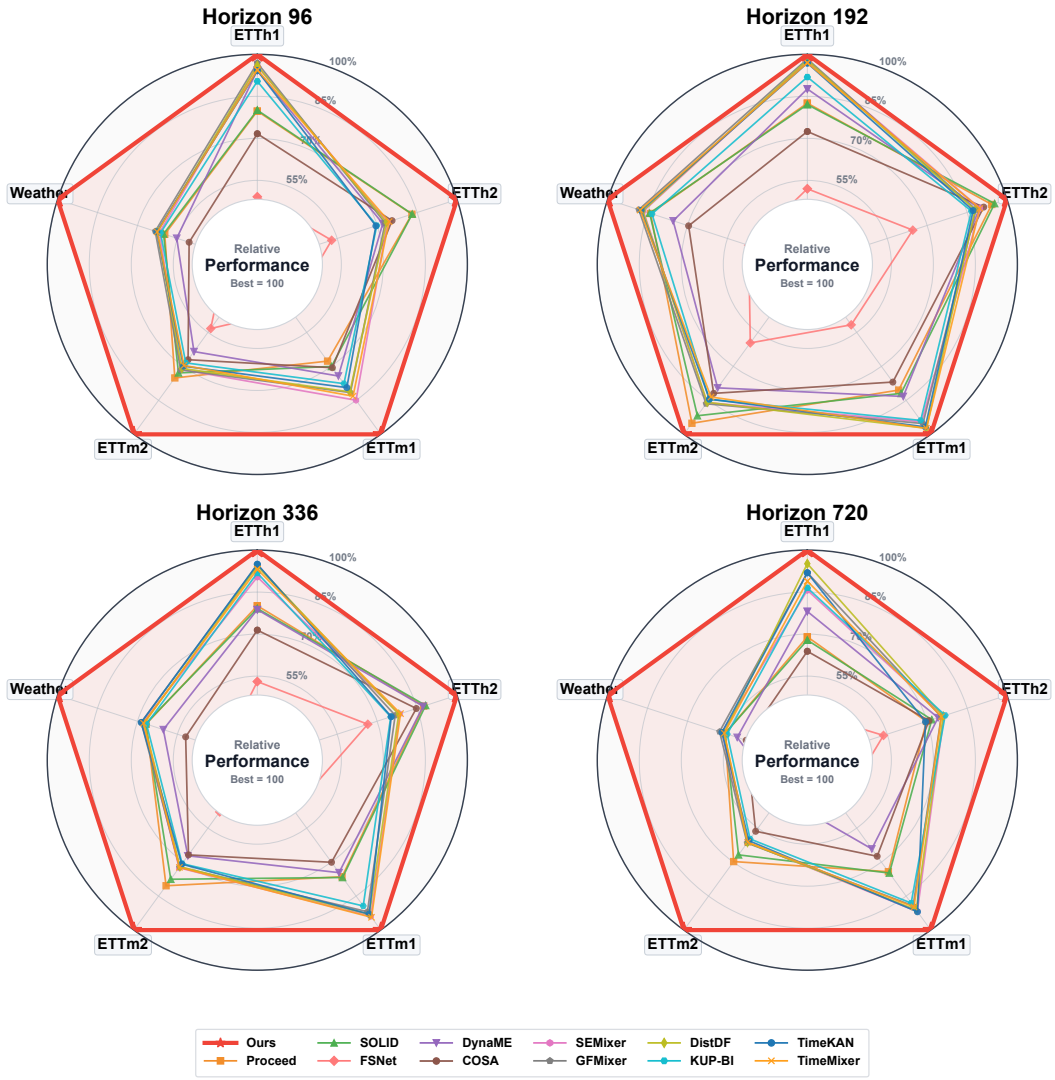

Figure 7: Comparison of prediction performance with an input length of 96.

The gains become even more pronounced as the forecasting horizon increases. At horizon 720, **SHIFT** reduces MSE by **29.3%** on ETTh2, **38.2%** on ETTm2, and up to **50.2%** on Weather over the strongest competing methods. The corresponding MAE reductions reach **17.2%**, **22.3%**, and **33.5%**, respectively. These substantial gains highlight the particular effectiveness of progressive feedback in long-horizon forecasting, where newly revealed observations provide increasingly valuable information for correcting deviations in the remaining trajectory.

Compared with online adaptation methods such as Proceed, SOLID, FSNet, DynaME, and COSA, **SHIFT** exhibits a consistently stronger and more balanced performance profile across datasets and horizons. It also remains highly competitive against recent forecasting models, including SEMixer, GFMixer, DistDF, KUP-BI, TimeKAN, and TimeMixer. Notably, the advantage of **SHIFT** is sustained rather than diminished as the prediction horizon grows, demonstrating that progressively incorporating newly available observations can effectively refine unreleased predictions and improve forecasting accuracy across both short- and long-term scenarios.

Table 4: Sample-level improvement and worsening rates for the same update decomposition. Rates are percentages of test windows whose next-block MSE is strictly lower or higher than the no-update MSE, respectively. Ties count toward neither rate.

Dataset	Target	Filtering only		Assimilation only		Full update	
		Improved	Worsened	Improved	Worsened	Improved	Worsened
Weather	B_2	72.21	27.79	87.42	12.58	87.26	12.74
	B_3	70.87	29.13	87.78	12.22	87.73	12.27
	B_4	73.26	26.74	87.59	12.41	87.56	12.44
CzeLan	B_2	76.48	23.52	78.10	21.90	78.10	21.90
	B_3	72.04	27.96	86.02	13.98	86.02	13.98
	B_4	76.23	23.77	85.76	14.24	85.76	14.24

C.3 Sample-Level Effects of Feedback Updates

Beyond average MSE reduction, we further examine whether the benefits of feedback updates are broadly distributed across individual test windows. Table 4 reports the proportion of samples whose next-block MSE is improved relative to the corresponding no-update prediction. Across all datasets and target blocks, feedback updates improve the majority of test windows, showing that the observed gains are not driven by only a small number of favorable cases.

The effect of assimilation is particularly evident on Weather and CzeLan. On Weather, assimilation improves approximately 87% of test windows across all three target blocks, compared with about 71–73% for filtering. On CzeLan, the improvement rate of assimilation reaches 86.02% and 85.76% on B_3 and B_4, respectively. Full update closely follows assimilation alone, further supporting trajectory revision as the dominant mechanism for exploiting newly revealed feedback.

Overall, these results show that the gains from assimilation are broadly distributed across test samples, rather than arising from a small subset of cases, and further confirm its central role in feedback-driven prediction refinement.

C.4 Comparison with Alternative Feedback-Revision Strategies

We further compare **SHIFT** with two alternative strategies for exploiting newly revealed feedback. **Full SHIFT** maintains $M = 8$ candidate trajectories and updates them after each feedback block: **Filtering** adjusts the hypothesis weights, while **Assimilation** revises the remaining trajectories.

MSD rolling uses the same multi-scale decomposition backbone (MSD) as SHIFT but adopts a rolling re-forecasting strategy. After each feedback block is revealed, the observed block is appended to the history, only the most recent $L = 336$ observations are retained, and MSD re-predicts the remaining future without updating model parameters. **MSD residual correction** instead generates a complete forecast once and, after receiving feedback, computes the mean prediction residual for each variable on the revealed block and adds it to subsequent predictions. This provides a simple parameter-free correction baseline.

As shown in Figure 8, Full SHIFT consistently achieves the best performance across both datasets and different forecasting horizons. Compared with **MSD rolling**, the advantage of Full SHIFT shows that simply appending newly observed feedback and re-forecasting the remaining future is not sufficient to fully exploit the information contained in feedback. The gap is more pronounced against **MSD residual correction**, indicating that feedback errors cannot be adequately handled by applying a simple additive residual adjustment to future predictions. Overall, the results demonstrate that explicitly learning how revealed observations should revise future trajectories provides more effective and consistent adaptation than conventional rolling re-forecasting or heuristic residual correction.

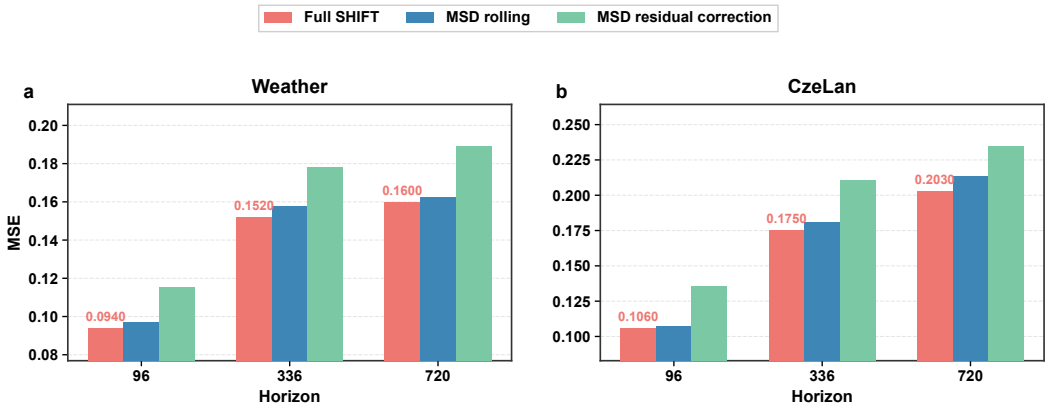

Figure 8: Comparison with alternative feedback-Revision strategies

C.5 Comparison with Rolling-Window Forecasters

In Table 1, the standard forecasting baselines follow the one-shot protocol and do not access observations revealed within the forecast horizon. To assess whether the advantage of **SHIFT** persists under matched feedback availability, we equip iTransformer and KUP-BI with a rolling-window wrapper that receives feedback at the same block boundaries as **SHIFT**. Each forecaster is trained once with $L = 336$ and $H = 720$, and its parameters remain fixed at test time. The horizon is divided into $K = 8$ blocks with a nominal length of $S = 96$, with the final block truncated to the remaining horizon. For block $\mathcal{B}_k$, the forecaster takes the most recent L observations, including all previously revealed blocks, as input and commits the first $|\mathcal{B}_k|$ steps of its forecast as the prediction for that block. The committed blocks are concatenated and evaluated on the same test windows as in the one-shot setting.

We further evaluate COSA and FSNet under the same feedback protocol. Test windows are processed in temporal order, and each method's online component is updated only using ground truth already observed when a forecast is issued. For a window whose horizon is partially revealed, the update loss is restricted to the revealed prefix. Forecasts are then committed block by block using the same rolling procedure. COSA applies its official output adapter and adaptation settings to the trained iTransformer, while FSNet uses its official network trained on the same data split. Restricting updates to observed targets prevents the future-information leakage that would arise from using the full horizon for an update immediately after issuing a forecast.

As shown in Table 5, **SHIFT** achieves the lowest MSE and MAE across all evaluated datasets, consistently outperforming rolling iTransformer, rolling KUP-BI, COSA, and FSNet. These gains under matched feedback availability suggest that the advantage of **SHIFT** extends beyond access to within-horizon observations. COSA does not consistently improve upon its rolling iTransformer backbone, suggesting that output-space adaptation offers limited additional benefit when the backbone already re-forecasts from the revealed blocks. The results therefore support the effectiveness of **SHIFT** relative to both rolling re-forecasting and the evaluated online adaptation methods. Moreover, the rolling wrapper repeatedly executes the complete forecaster on updated input windows, whereas **SHIFT** encodes the history only once and incorporates each revealed block into compact latent.

C.6 Joint Sensitivity of the Loss Weights

We investigate the interaction between the progressive prediction loss and the hindsight bridge loss. The relevant SHIFT objective is

$$\mathcal{L}_{\text{SHIFT}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{prog}}\mathcal{L}_{\text{prog}} + \lambda_{\text{bridge}}\mathcal{L}_{\text{bridge}}, \tag{17}$$

where $\mathcal{L}_{\text{pred}}$ supervises the initial prediction, $\mathcal{L}_{\text{prog}}$ optimizes progressively revised predictions after feedback assimilation, and $\mathcal{L}_{\text{bridge}}$ transfers corrective information from hindsight teachers. Thus, λ_{prog} controls stage-wise revision, while λ_{bridge} controls teacher-guided supervision.

Table 5: Comparison with rolling-window forecasters on the $L = 336$, $H = 720$ setting. All methods receive feedback at the same block boundaries as **SHIFT** ($S = 96$, $K = 8$). The best results are highlighted in **red** and the second-best results in blue.

Dataset	iTransformer + Rolling		KUP-BI + Rolling		COSA + Rolling		FSNet + Rolling		**SHIFT**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	0.198	0.245	0.233	0.298	0.188	0.274	0.203	0.270	**0.160**	**0.217**
ETTm2	0.203	0.299	0.273	0.321	0.225	0.309	0.261	0.335	**0.183**	**0.274**
Exchange	0.221	0.353	0.141	0.401	0.221	0.354	0.111	0.247	**0.102**	**0.227**

We perform a joint grid search on ETTh1 while keeping the architecture, data split, feedback protocol, training budget, and checkpoint selection unchanged. Each configuration is trained independently and evaluated using deployed MSE and MAE.

As shown in Figure 9, the results show that λ_{prog} is the dominant sensitivity direction. Excessively large values overemphasize intermediate revisions and degrade both metrics, whereas very small values weaken supervision of the correction process. Overall, relatively weak progressive weighting provides the best balance between initial forecasting accuracy and subsequent revision.

The effect of λ_{bridge} is milder and metric-dependent. Stronger bridge supervision can improve MSE when progressive supervision is weak, but excessive weighting may overly constrain candidate trajectories and slightly worsen MAE. Conversely, insufficient bridge supervision provides limited guidance for organizing alternative futures. The different MSE- and MAE-optimal regions therefore reflect a trade-off between teacher correction and trajectory flexibility.

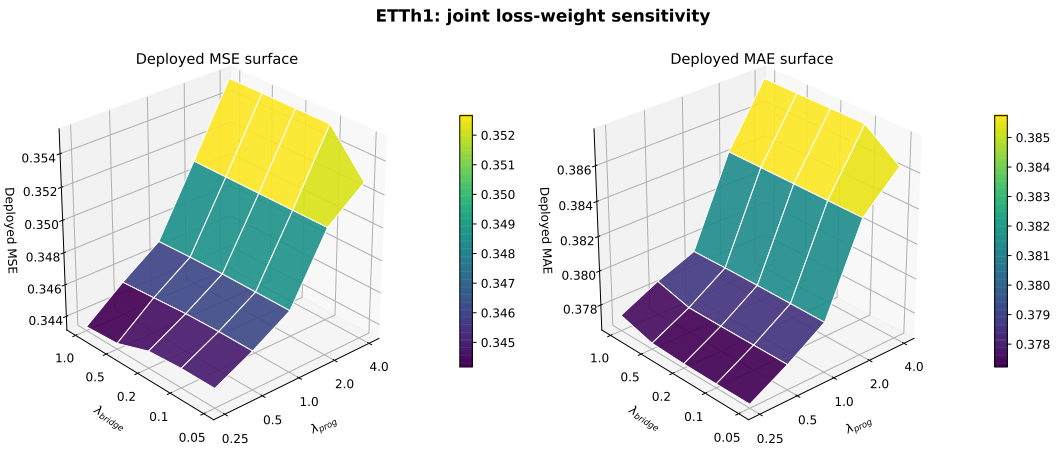

Figure 9: Hyperparameter sensitivity experiments.

C.7 Qualitative Analysis of Feedback-Driven Trajectory Revision

Figure 10 visualizes two Weather cases with four hypotheses. In both cases, the initial prediction deviates substantially from the ground truth. As each feedback block becomes available, SHIFT assimilates the revealed observations and rewrites the remaining trajectory. For air density, the successive revisions progressively raise the forecast toward the increasing target pattern. For maximum wind speed, feedback corrects the initially overestimated level and improves the prediction over subsequent unrevealed intervals. Because each revision only uses observations to the left of the corresponding feedback boundary, these examples illustrate how SHIFT causally adjusts its forecast during deployment.

The belief weights remain close to a uniform distribution, and the four candidate trajectories largely overlap. Therefore, the improvements in these cases mainly arise from feedback assimilation and trajectory rewriting rather than strong hypothesis selection. Together with the more differentiated

belief evolution observed under a larger hypothesis bank, these results suggest that a small bank can still support effective online correction, while additional hypotheses provide more capacity for representing distinct futures and reallocating belief after feedback.

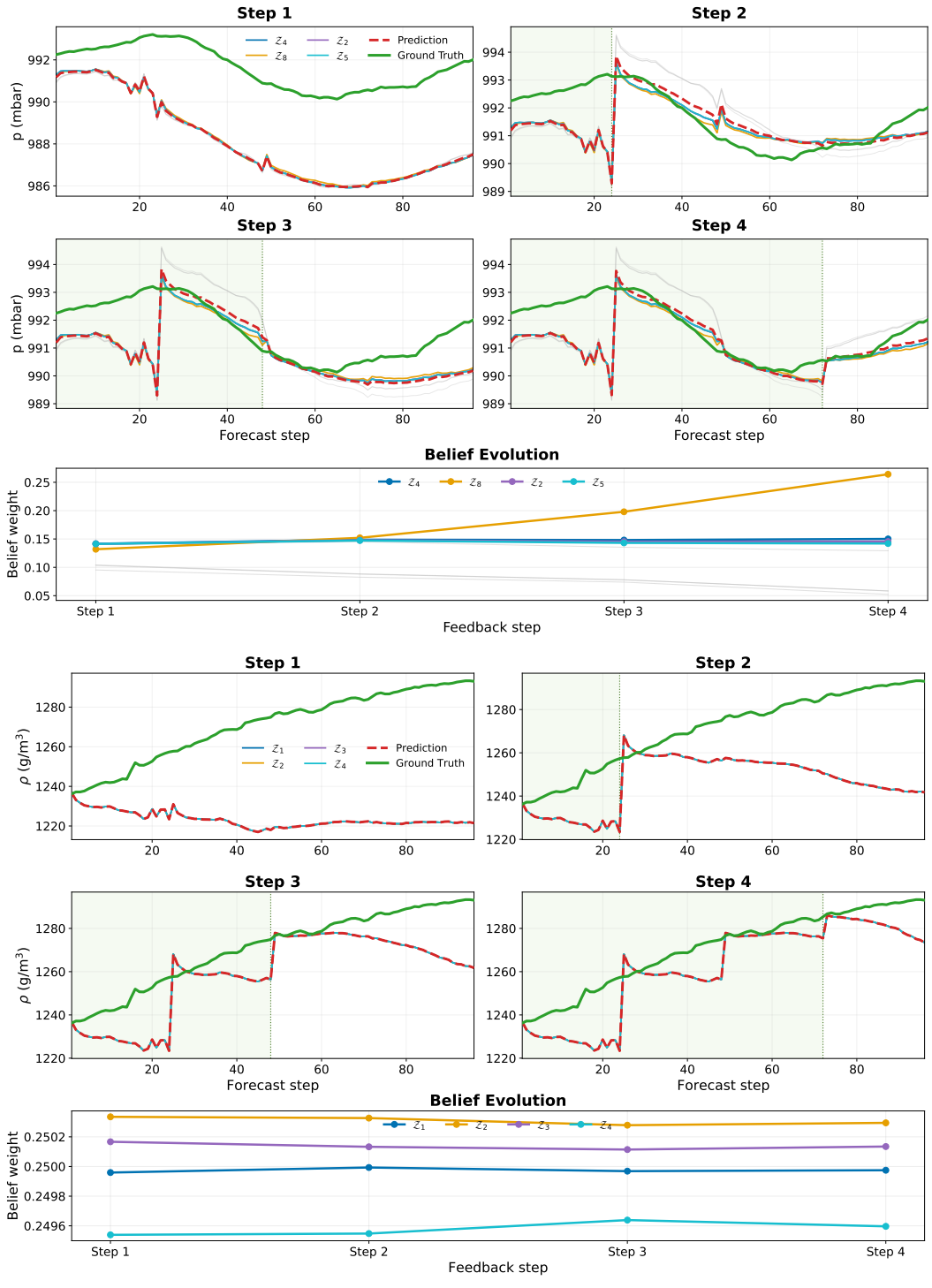

Figure 10: Prediction visualization on the Weather dataset under two representative cases.

Table 6: Backbone generality of SHIFT under an input length of $L = 336$. For each forecasting backbone, we compare the vanilla model with its SHIFT-augmented counterpart. Lower MSE and MAE indicate better performance. Bold denotes the better result within each vanilla/SHIFT backbone pair.

Dataset	Backbone	Variant	$H = 96$		$H = 192$		$H = 336$		$H = 720$	
			MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	DLinear	Vanilla	0.3680	0.3870	0.4070	0.4120	0.4460	0.4380	0.4730	0.4870
		+ SHIFT	**0.3480**	**0.3816**	**0.3656**	**0.3970**	**0.3856**	**0.4089**	**0.3767**	**0.4204**
	iTransformer	Vanilla	0.3860	0.4050	**0.4410**	**0.4360**	0.4870	0.4580	0.5030	0.4910
		+ SHIFT	**0.3818**	**0.3987**	0.4741	0.4397	**0.4479**	**0.4339**	**0.3953**	**0.4326**
ETTm2	DLinear	Vanilla	0.1650	0.2480	0.2220	0.2870	0.2760	0.3290	0.3840	0.4020
		+ SHIFT	**0.0992**	**0.2010**	**0.1900**	**0.2758**	**0.2012**	**0.2879**	**0.2145**	**0.2992**
	iTransformer	Vanilla	0.1800	0.2640	0.2500	0.3090	0.4120	0.3480	0.3110	0.4070
		+ SHIFT	**0.1084**	**0.2132**	**0.1758**	**0.2706**	**0.1823**	**0.2741**	**0.1891**	**0.2819**

C.8 Backbone Generality of SHIFT

To examine whether the effectiveness of SHIFT depends on a particular forecasting architecture, we apply SHIFT to two substantially different backbones: the lightweight linear model DLinear and the Transformer-based iTransformer. For each backbone, we compare its vanilla forecasting performance with the corresponding SHIFT-augmented variant under the same input length of $L = 336$. The SHIFT variants use a feedback block length of $S = 24$.

As shown in Table 6, SHIFT consistently improves DLinear across all eight dataset–horizon configurations. For iTransformer, SHIFT reduces both MSE and MAE in seven out of eight configurations, with the only exception occurring on ETTh1 at $H = 192$. Overall, SHIFT improves both error metrics in 15 out of 16 backbone–dataset–horizon configurations. The consistent improvements across both a lightweight linear model and a Transformer-based model demonstrate that the effectiveness of SHIFT is not tied to a specific forecasting architecture, supporting its general applicability as a backbone-agnostic progressive forecasting framework.

D Limitations

While SHIFT demonstrates consistent improvements across diverse forecasting benchmarks, several aspects remain open for further investigation. First, the current formulation assumes that observations within the forecast horizon become available progressively at predefined feedback boundaries. Although this setting naturally arises in many operational forecasting scenarios, practical systems may provide feedback at irregular intervals or with varying delays. Extending SHIFT to asynchronous or adaptive feedback schedules would broaden its applicability.

Second, the current hindsight teacher defines its target distribution using an MSE-based discrepancy between candidate trajectories and the realized future. This simple formulation provides stable supervision but mainly emphasizes point-wise amplitude differences. Incorporating richer trajectory-aware criteria, such as temporal-shape or frequency-sensitive similarity, may provide more informative supervision when candidate futures primarily differ in temporal structure.

AI Use Statement

AI tools were used to assist with language polishing and rewriting, provide suggestions on the organization and structure of the manuscript, improve the presentation of the model framework in Figure 2, and refactor portions of the code to enhance readability and maintainability. All final decisions regarding the manuscript content, figures, and code were made and verified by the authors.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we make the source code for **SHIFT** publicly available in an anonymous repository[1]. The main paper describes the model architecture, training objective, and causal predict-then-reveal protocol. Section 5 reports the datasets, training hyperparameters, validation-based early stopping, and computing hardware.

[1] https://anonymous.4open.science/r/SHIFT-FOR-ICLR-D8C9/README.md