Beyond Uniform Updates: Drift Pattern Aware Online Time Series Forecasting under Delayed Feedback

Xingwang Li[1] , **Fei Teng**[1,2] , **Cong Zhou**[3] and **Qiang Duan**[4]
[1]School of Computing and Artificial Intelligence, Southwest Jiaotong University
[2]Engineering Research Center of Sustainable Urban Intelligent Transportation, Ministry of Education
[3]Advanced Institute for Ocean Research, Southern University of Science and Technology, Shenzhen
[4]Information Sciences and Technology Department, The Pennsylvania State University
lixingwang@my.swjtu.edu.cn, fteng@swjtu.edu.cn, zhouc@sustech.edu.cn, qxd2@psu.edu

Abstract

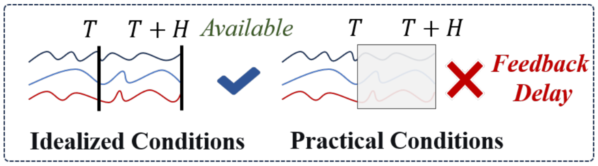
Online time series forecasting relies on continual updates to cope with concept drift. In multi-step forecasting, however, the ground truth for an H-step prediction arrives only after H steps, so a delayed residual entangles persistent drifts with transient shocks and seasonal fluctuations. Existing methods typically apply a uniform update rule to all delayed errors, which can overreact to noise and under-adapt to real drift. We view delayed residuals as compressed observations of latent drift over the horizon, and propose PADRE, a drift evidence driven framework that converts each delayed feedback event into a context-sensitive adaptation decision. PADRE builds a Tri-View drift representation from (i) the current input context, (ii) recent delayed-residual trajectories, and (iii) retrieved typical types of drift, and quantifies their agreement via a geometric consistency measure. To exploit recurring drift structure, PADRE mines an offline Pattern Bank of regime-conditioned residual prototypes and retrieves a soft pattern prompt online. A lightweight prompt guided policy outputs a continuous update gate that scales the backbone's gradient step, enabling decisive updates under coherent evidence and conservative updates otherwise. Experiments on five real-world benchmarks and three backbones demonstrate consistent gains under delayed feedback, reducing MSE by 6–15% relative to recent baselines and improving update stability across horizons.

1 Introduction

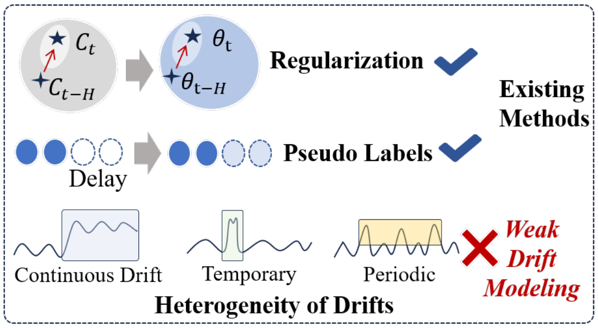
Time series forecasting underpins a wide range of real-time services, from energy allocation and traffic control to financial trading and industrial monitoring [Wang *et al.*, 2025b; Li *et al.*, 2026a; Wu *et al.*, 2025; Li *et al.*, 2026b]. Classical forecasting pipelines typically assume that a stationary or slowly varying data generating process can be captured and updated only sporadically. However, modern cyber physical systems interact with users, policies, and environments that change continuously, inducing rich forms of concept drift in both the marginal dynamics and the input–output relationships [Demirel and Holz, 2025; Na *et al.*, 2025; Li *et al.*, 2025]. As a result, a forecaster that remains faithful to its initial training distribution can rapidly become mis-calibrated, systematically under-reacting to emerging regimes or amplifying outdated patterns into costly decisions [Liu *et al.*, 2025; Dama *et al.*, 2025; Wang *et al.*, 2025a; Brenner *et al.*, 2025].

Online time series forecasting (OTSF) has therefore attracted attention for combating concept drift by updating models directly on streaming data [Gupta *et al.*, 2025; Zhao *et al.*, 2025; Lee *et al.*, 2025; Cai *et al.*, 2025]. Rather than relying on batch retraining, an OTSF learner interleaves prediction and adaptation, using recent observations to track evolving regimes and maintain performance over long horizons. Theoretically, such continual adaptation resolves the stability plasticity dilemma: enabling the model to adjust rapidly to structural shifts. Real deployments, however, rarely provide "instant" supervision. In multi-step forecasting, a prediction issued at time t targets the entire horizon $[t, t + H]$, while its ground-truth outcome is only revealed after the horizon elapses (i.e., at $t + H$), as illustrated in Fig 1. This delayed-feedback protocol creates a temporal gap between the decision point and its evaluation: the learner adapts using delayed residuals that are temporally aggregated over $[t, t + H]$, inevitably entangling persistent regime shifts with periodicity and short lived shocks.

Recent OTSF methods progress along two lines. The first line improves the exploitation of recent inputs—e.g., via ensemble mechanisms or context aware sample selection as in OneNet [Wen *et al.*, 2023] and SOLID [Chen *et al.*, 2024], but it overlooks a fundamental constraint of multi-step forecasting: each H-step decision is not observable when the decision is made, and only becomes known after the forecast horizon has passed. The second line acknowledges the H-step feedback delay and updates the model through drift estimation [Zhao and Shen, 2025] or pseudo label supervision [Lau *et al.*, 2025] (Fig.1(b)). However, a limitation persists across current SOTA: adaptation is governed by uniform update rules, where delayed errors trigger comparable gradient steps regardless of the underlying drift type. Because the delayed residual compresses all changes over $[t, t + H]$ into a temporally aggregated signal, such one-size-fits-all updates conflate systematic shifts [Shao *et al.*, 2025; Dong *et al.*, 2024; Wu *et al.*, 2026] that warrant adaptation, transient outliers ,



(a) H-step Delayed Supervision in Online Multi-step Forecasting.



(b) Existing Methods Under Feedback Delay: Effective but Drift-Type-Agnostic Updates.

Figure 1: Challenges of online forecasting under delayed feedback.

and recurring periodic patterns.

Under H-step delayed feedback, we argue that the supervision signal should be reinterpreted. The delayed residual, $r_t = \hat{Y}_t - Y_t$, serves not as an informative training target, but as a noisy and temporally aggregated observation of the latent drift states over the horizon $[t, t+H]$. Accordingly, our goal is to make online updates selective and stable by treating r_t as evidence of what has accumulated over $[t, t+H]$, rather than using it as raw supervision. Within these constraints, we contextualize delayed residuals via three complementary cues: the current input context (regime transition), the residual trajectory (error evolution), and recurring drift patterns from past events (shift matching). The tripartite perspective provides a structured view of drift signals over the delay window and informs the update strategy.

Building on this tripartite perspective, we propose PADRE, an online forecasting framework, converting each delayed feedback event into a structured adaptation decision. PADRE first constructs a Tri-View drift representation from the current context, the residual trajectory, and retrieved drift memories, and quantifies their agreement via a geometric consistency score to separate coherent drift evidence from transient or poorly explained fluctuations. On top of this representation, an offline-mined Pattern Bank summarizes recurring regime–residual configurations; online retrieval provides a soft pattern prompt that relates the current event to previously observed shifts. Finally, a gated update policy, guided by the drift evidence and the pattern prompt, outputs a continuous update gate that modulates the backbone's effective learning rate—enabling decisive updates when evidence is consistent and conservative updates otherwise.

Our main contributions are threefold:

- We interpret delayed residuals as noisy observations of latent drift states rather than direct supervision. This leads to a Tri-View drift representation that geometrically fuses online context, backward residual evolution, and episodic memory, providing a stable space for reasoning about when and how to adapt.
- We propose a unified, evidence-driven adaptation framework PADRE, integrates (i) a Tri-View Drift Module with geometric consistency scoring, (ii) a Pattern Bank distilling reusable regime–residual prototypes with soft retrieval, and (iii) a gated policy enabling adaptive yet stable online learning under delayed feedback.
- Extensive experiments on five real-world benchmarks and three strong backbone forecasters demonstrate that our framework consistently improves MSE and MAE under delayed feedback, achieving substantial relative gains over recent OTSF baselines. Ablation and analysis further verify that both the Tri-View geometry and the pattern-bank-guided gating are effective for stable, sample-efficient adaptation.

2 Related Work

Concept drift is a primary source of performance degradation in real world time series forecasting, where the underlying dynamics and input–output relations evolve over time [Witulska *et al.*, 2026; He *et al.*, 2025; Zhang *et al.*, 2025]. OTSF tackles this challenge by interleaving prediction and adaptation on streaming data. We organize related work by whether the method explicitly models the delayed supervision protocol inherent in multi-step forecasting.

2.1 Classic Online Time Series Forecasting

A common strategy in classic OTSF is to strengthen short term exploitation of recent context via architectural mechanisms, ensembling, or lightweight memories. FSNet [Pham *et al.*, 2023] and OneNet [Wen *et al.*, 2023] improve robustness under non-stationarity by incorporating associative memory to capture evolving patterns. Another direction performs context-aware sample selection to decide when to adapt, exemplified by SOLID [Chen *et al.*, 2024] (and its variants used in practice). LEAF [Chen *et al.*, 2025] learns adaptation strategies via meta-learning by separating drift effects at different temporal granularities.

These methods form strong baselines for online adaptation under drift; however, they typically do not treat delayed supervision as a first-class constraint in multi-step forecasting. When the accuracy of an H-step decision is only observable after the horizon elapses, update signals may become misaligned with the true delayed objective, and adaptation can be difficult to calibrate using proxy cues alone.

2.2 Online Forecasting with Delayed Feedback

A second line of work explicitly acknowledges H-step feedback delay. Proceed [Zhao and Shen, 2025] bridges the temporal gap by proactively estimating drift so that the model can adapt despite delayed supervision. DSOF [Lau *et al.*, 2025] adopts a dual stream framework that decouples slow experience replay from fast temporal adaptation to mitigate information leakage while coping with delayed feedback.

Despite respecting the feedback delayed protocol, existing methods commonly rely on uniform update rules: delayed residuals trigger comparable gradient steps regardless of the underlying drift type. This can be problematic because a delayed residual aggregates changes accumulated over $[t, t + H]$, conflating persistent regime shifts, transient shocks and recurring periodic variations. Moreover, delay aware methods typically treat each delayed-feedback episode in isolation, without distilling recurring regime–residual structures into reusable priors for calibrating update strength. Our work advances this delay-aware line by enabling selective, multi-view grounded updates under feedback delayed.

3 Methodology

3.1 Problem Definition

We consider online multi-step time-series forecasting under delayed supervision. At each time step t, the model observes a look-back window $x_t \in \mathbb{R}^{L\times V}$, where L is the lookback length and V is the number of variables, and produces an H-step-ahead forecast $\hat{y}_t = f_\theta(x_t) \in \mathbb{R}^{H\times V}$. The true target y_t becomes available only after a delay of H-steps, i.e., at time $t+H$. Consequently, when the error $r_t = \hat{y}_t - y_t$ is observed, it reflects the cumulative effect of all changes that happened over the horizon $[t, t + H]$, including noise, seasonal fluctuations, and concept drift. Naively updating the model with every delayed error is therefore problematic: it can be overly conservative (under-reacting to real drift) or aggressive (overfitting benign fluctuations).

3.2 Framework Overview

Our framework consists of four coupled modules (Fig. 2): (i) a backbone forecaster f_θ that produces multi-step predictions; (ii) a *Pattern Bank* $\mathcal{P}$ learned offline, capturing recurring regimes and the backbone's typical residual shapes under those regimes; (iii) a *Tri-View Drift Module* that encodes delayed feedback into three complementary drift views and a geometric consistency score; and (iv) a *Prompt-Guided Update Policy* π_ϕ that queries $\mathcal{P}$ online and outputs a continuous update gate $\gamma_t \in (0, 1)$ to modulate the online gradient step.

Offline stage. We first pretrain the backbone f_θ on historical data. Using the pretrained backbone, we construct the Pattern Bank $\mathcal{P}$ as a compact set of regime-conditioned residual prototypes (Sec. 3.3). $\mathcal{P}$ serves as a contextual prior during online forecasting. Moreover, to remember unseen drift patterns beyond those captured in the offline Pattern Bank, PADRE continuously updates an online episodic memory M with high impact drift episodes (Sec. 3.6).

Online stage. At time t, the forecaster observes the look-back window x_t and outputs $\hat{y}_t = f_\theta(x_t)$ and caches it as y^b_t for residual construction. When the target y_t arrives at time $t + H$, we treat it as feedback for the earlier decision point t and compute the delayed residual r_t. Using (x_t, r_t), we construct tri-view drift evidence $(z^{on}_t, z^{back}_t, z^{mem}_t)$—the online, backward, and memory views defined in Sec. 3.4 and a geometric consistency score A_t, then query the offline Pattern Bank p to obtain a pattern prompt p_t. The update policy π_ϕ maps p_t, A_t to a continuous gate $\gamma_t \in (0, 1)$, which scales the backbone gradient step on the delayed sample (recomputing $\widetilde{y}_t = f_\theta(x_t)$) for an up-to-date loss).

3.3 Pattern Bank Training

Empirically, we observe that the backbone exhibits systematic, regime-dependent error patterns: when the system operates under similar temporal regimes (e.g., workday morning peak, off-peak night, seasonal transition), it tends to make errors with similar shapes and biases. We distill this regularity into a compact Pattern Bank, where each prototype jointly encodes (i) a regime embedding and (ii) the backbone's typical residual behavior under that regime.

Regime prototypes. For each training window x_t, we compute a regime embedding

$$q^{reg}_t = f_{reg}(\mathrm{vec}(x_t)), \tag{1}$$

where f_{reg} is a lightweight projection head (e.g., a shallow MLP), and $\mathrm{vec}(\cdot)$ is a parameter-free vectorization. We apply K-means clustering in the regime space to obtain cluster centers $\{z^{regime}_k\}^K_{k=1}$ and assignments $\{C_k\}^K_{k=1}$, where C_k denotes the set of windows assigned to regime k. Each z^{regime}_k summarizes a recurrent input regime.

Regime-conditioned residual prototypes. Rather than clustering residuals independently, we condition residual prototypes on regimes. For each regime cluster C_k, we run the pretrained backbone to compute residuals

$$r_t = f_\theta(x_t) - y_t, \quad t \in C_k. \tag{2}$$

We map each residual segment through a residual encoder f_{res},

$$q^{res}_t = f_{res}(\mathrm{vec}(r_t)), \tag{3}$$

and aggregate them within the cluster:

$$z^{residual}_k = \frac{1}{|C_k|} \sum_{t \in C_k} q^{res}_t. \tag{4}$$

The vector $z^{residual}_k$ captures the typical error shape that the backbone tends to make when the system is in regime k, serving as a regime-specific error label.

Pattern prototypes. Finally, each pattern prototype in the bank concatenates its regime and residual components:

$$p_k = \left[z^{regime}_k \parallel z^{residual}_k\right], \quad k = 1, \ldots, K, \tag{5}$$

yielding the Pattern Bank $\mathcal{P} = \{p_k\}^K_{k=1}$. During online adaptation, queries derived from the current window and its delayed residual are matched against $\mathcal{P}$ via attention, and the resulting pattern prompt informs the update policy about (i) which regime the system is currently in and (ii) how the backbone has updated in that regime. This regime-conditioned design avoids arbitrary pairing between input and residual clusters, acting as an amortized prior for stable online updates.

3.4 Tri-View Drift Representation

Under delayed feedback, a single residual vector is insufficient to characterize drift: it conflates noise with distributional changes and ignores historical context. We therefore construct a Tri-View Drift Module that encodes delayed feedback from three complementary perspectives: an *online view* based on the input context that produced the cached forecast, a *backward view* based on the recent evolution of delayed residuals, and a *memory view* based on past drift episodes that previously triggered strong adaptation.

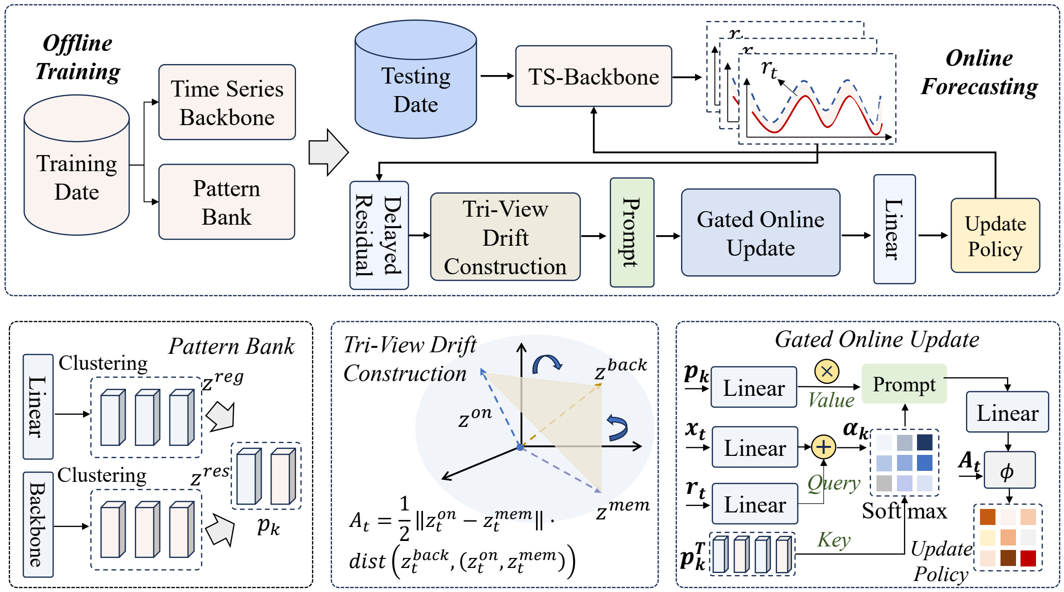


Figure 2: Overview of PADRE. Top: the end-to-end workflow, where an offline stage (top-left) pretrains the backbone and constructs the Pattern Bank, and an online stage (top-right) performs delayed-feedback forecasting and gated updates. Bottom: module details of Pattern Bank, tri-view drift construction, and the prompt guided gated online update policy.

Online view. We encode the input window x_t that generated the cached prediction y_t^b (i.e., the H-step prediction issued at time t) as

$$z_t^{on} = g_{on}(\mathrm{vec}(x_t)) \in \mathbb{R}^D, \tag{6}$$

where g_{on} is a two-layer MLP. This view captures drift cues directly visible from the current context (e.g., entering a new daily regime).

Backward view. When the delayed target y_t arrives, we compute the delayed residual $r_t = y_t^b - y_t$ and append it to $\mathcal{R}$. The buffer maintains the most recent M_r residuals in chronological order. We form the backward summary by concatenation

$$r^* = [r_{t-M_r+1};\ r_{t-M_r+2};\ \dots;\ r_t], \tag{7}$$

and compute the backward view as

$$z_t^{back} = g_{back}(\mathrm{vec}(r^*)) \in \mathbb{R}^D, \tag{8}$$

where g_{back} is a two-layer MLP. The chronological concatenation preserves temporal ordering, enabling g_{back} to capture how prediction errors have evolved (e.g., increasing, stable, or decaying). Specifically, the residual trajectory provides a *posterior trace* of the changes accumulated over the delayed horizon $[t, t+H]$, complementing the context anchor z_t^{on}.

Memory view. The drift memory $\mathcal{M}$ stores backward-view embeddings from past update steps that were deemed high-impact by the policy. Given the current backward view z_t^{back}, we retrieve the closest memory vector by cosine similarity and project it as

$$\tilde{m}_t = \arg\max_{m\in\mathcal{M}} \cos(m, z_t^{back}), \qquad z_t^{mem} = g_{mem}(\tilde{m}_t) \in \mathbb{R}^D, \tag{9}$$

where g_{mem} is a small MLP. If $\mathcal{M}$ is empty, we fall back to $z_t^{mem} = z_t^{back}$.

Remark (Pattern Bank $\mathcal{P}$ vs. drift memory $\mathcal{M}$). $\mathcal{P}$ is an *offline* prototype library distilled from historical training data; it stores regime–residual *prototypes* $\{p_k\}$ and is queried online to provide a contextual prior (a pattern prompt) for update decisions. In contrast, $\mathcal{M}$ is an *online* episodic buffer that stores a small set of *past drift episodes* (backward-view embeddings) that previously triggered strong adaptation; it is used only to construct the memory view z_t^{mem} for retrieval and multi-view consistency diagnosis.

3.5 Multi-View Consistency via Triangle Area

We treat the backward view z_t^{back}, derived from delayed residual evolution, as a relational anchor that links the online context z_t^{on} and the episodic memory z_t^{mem}. Our key inductive bias is as follows.

Assumption 1. *When forecasting errors are caused by a systematic concept shift, it induces coherent changes in (i) the current input context, (ii) the delayed-residual evolution, and (iii) the resemblance to previously observed high-impact drift episodes when such episodes exist in the episodic memory M.* Accordingly, the tri-view provides complementary estimates of the same latent drift direction and tends to be mutually compatible in representation space; the memory view serves only as an auxiliary recurrence prior rather than a mandatory match.

By contrast, errors stemming from measurement noise, outliers, or unseen drift mechanisms yield inconsistent multi-view signals. We therefore quantify geometric consistency among the three views [Liberti *et al.*, 2014]. Define

$$u_t = z_t^{back} - z_t^{on}, \qquad v_t = z_t^{back} - z_t^{mem}. \tag{10}$$

We compute the triangle area by the Gram-determinant form:

$$A_t = \frac{1}{2}\sqrt{\|u_t\|_2^2\|v_t\|_2^2 - (u_t^\top v_t)^2}. \tag{11}$$

Geometrically, A_t measures how far the backward view deviates from the line connecting the context and memory views: it vanishes when the three points are collinear and grows as they diverge. A small A_t indicates that the three views provide a coherent explanation of the delayed feedback episode (e.g., a familiar context with residual evolution consistent with past high-impact events). A large A_t signals disagreement across evidence sources, suggesting a noisy or atypical episode poorly explained by memory and context.

Note that A_t serves as a diagnostic soft consistency regularizer rather than a hard veto: even when the memory view is less informative (e.g., for unseen drifts), the policy can increase the update gate γ_t if doing so improves the post-update prediction loss, while the A_t-weighted term mainly discourages overly aggressive updates under conflicting evidence.

3.6 Prompt-Guided Gated Online Update

Once the delayed residual r_t is observed, we must decide how strongly to update the model parameters. Rather than using discrete rules to set the update magnitude, we frame this decision as a continuous gated policy that maps structured evidence to an update gate $\gamma_t \in (0, 1)$.

Pattern prompting. We first query the Pattern Bank with the current input–residual pair. A query embedding is constructed as

$$Q_t = \big[f_{reg}(\mathrm{vec}(x_t)) \parallel f_{res}(\mathrm{vec}(r_t))\big]. \tag{12}$$

Attention over the bank yields

$$\alpha_k(t) = \mathrm{softmax}\left(\frac{p_k^\top Q_t}{\tau}\right), \qquad p_t = \sum_{k=1}^{K} \alpha_k(t) p_k, \tag{13}$$

where τ is a temperature hyper-parameter. The resulting pattern prompt p_t serves as a soft label indicating which regime–residual patterns from the bank best match the current delayed feedback event.

Update policy. We form a state vector by concatenating (i) the pattern prompt p_t, (ii) the online drift view z_t^{on}, and (iii) the multi-view consistency score A_t:

$$s_t = \big[p_t \parallel z_t^{on} \parallel A_t\big]. \tag{14}$$

A lightweight policy network π_ϕ maps this state to a scalar update gate:

$$\gamma_t = \pi_\phi(s_t) = \sigma(\mathrm{MLP}(s_t)) \in (0, 1), \tag{15}$$

where σ is the sigmoid function. Intuitively, the policy should output large gates for informative, familiar drift patterns (e.g., confident pattern prompt and small A_t) and small gates for noisy or poorly explained episodes (e.g., large A_t or atypical prompts).

Gated online update. Given γ_t, we modulate the step size of the backbone update. Upon receiving the delayed target y_t, we recompute $\tilde{y}_t = f_\theta(x_t)$ and define the robust regression loss $L_{\mathrm{pred}}(t) = \mathrm{SmoothL1}(\tilde{y}_t, y_t)$. The backbone parameters are updated by a gated gradient step:

$$\theta \leftarrow \theta - \eta\, \gamma_t\, \nabla_\theta L_{\mathrm{pred}}(t), \tag{16}$$

where η is the base learning rate. We emphasize that the cached prediction y_t^b is used to form the delayed residual r_t for drift evidence, while $\tilde{y}_t$ is recomputed under the current parameters to provide an up-to-date training signal for the actual gradient step.

Episodic memory update. If $\gamma_t > \tau_{mem}$, we treat the current episode as a high-impact drift event and insert the corresponding backward-view embedding into $\mathcal{M}$ using the protocol in Sec. 3.4. (This threshold is used only for memory management, while the update magnitude is governed continuously by γ_t.)

Learning the update policy. To make π_ϕ trainable with a clear objective while avoiding second-order derivatives, we optimize the backbone and the policy in an alternating manner. When updating θ via Eq. (17), we treat γ_t as a constant (no gradient is propagated into ϕ). When updating ϕ, we minimize a one-step look-ahead surrogate that evaluates the post-update prediction loss using a first-order approximation. Let $g_t = \mathrm{stopgrad}(\nabla_\theta L_{\mathrm{pred}}(t))$ be the backbone gradient treated as a constant for policy learning, and define a virtual post-update parameter:

$$\theta_t^+(\gamma_t) = \theta - \eta\, \gamma_t\, g_t. \tag{17}$$

We then optimize the policy parameters by

$$L_\pi(t) = L_{\mathrm{pred}}(t; \theta_t^+(\gamma_t)) + \lambda_\pi\, \gamma_t\, \mathrm{stopgrad}(A_t), \tag{18}$$

and update ϕ via $\phi \leftarrow \phi - \eta_\phi \nabla_\phi L_\pi(t)$. The first term encourages gates that improve the post-update fit, while the second term discourages large updates when the tri-view evidence is inconsistent. We stop the gradient through A_t so that it remains a diagnostic statistic rather than an alignment target.

4 Experiment

4.1 Experimental Settings

Datasets and Baseline. We evaluate PADRE on five widely-used benchmarks covering diverse domains: ETTh2 and ETTm1 (energy), Weather (meteorology), ECL (electricity consumption), and Traffic (transportation). To simulate realistic online scenarios with concept drift, we follow a challenging chronological train-validation-test split of 20:5:75 [Zhao and Shen, 2025]. We compare against representative SOTA baselines from both **classic delay-agnostic OTSF** (OneNet, SOLID++, LEAF) and **OTSF under feedback delayed** (PROCEED, DSOF), ensuring comprehensive coverage of prior work. Following the setup in Proceed, we adopt the more performant SOLID++ variant, which continually fine-tunes parameters across the online phase, rather than the original SOLID method.

Model	Method \ Dataset / H	ETTh2 24	ETTh2 48	ETTh2 96	ETTm1 24	ETTm1 48	ETTm1 96	Weather 24	Weather 48	Weather 96	ECL 24	ECL 48	ECL 96	Traffic 24	Traffic 48	Traffic 96
TCN	\	3.156	4.141	6.019	1.770	1.908	1.673	3.301	9.718	8.588	57.782	57.948	58.729	0.569	0.637	0.676
	OneNet	2.965	4.892	8.257	1.335	1.547	1.040	0.861	1.182	1.484	9.757	10.847	15.932	0.462	0.544	0.591
	SOLID++	3.231	4.111	6.191	0.674	0.795	0.868	0.867	1.311	1.670	5.991	7.340	9.412	0.418	0.479	0.529
	LEAF	3.043	4.346	6.052	0.785	0.917	1.026	0.825	1.224	1.518	6.075	8.134	9.662	0.463	0.528	0.571
	DSOF	2.975	4.136	5.977	0.554	0.728	0.789	0.723	1.035	1.382	5.864	7.173	9.261	0.424	0.457	0.526
	PROCEED	2.908	4.056	5.891	0.531	0.704	0.780	0.707	0.959	1.314	5.907	7.192	9.183	0.413	0.454	**0.511**
	Our	**2.753**	**3.824**	**5.543**	**0.446**	**0.616**	**0.673**	**0.624**	**0.836**	**1.196**	**5.684**	**6.946**	**8.957**	**0.404**	**0.442**	0.515
PatchTST	\	2.880	4.103	6.178	0.549	0.726	0.769	0.746	1.006	1.311	4.207	4.942	6.042	0.378	0.386	0.403
	OneNet	2.717	4.095	6.044	0.681	0.721	0.788	0.824	1.052	1.319	4.111	4.752	5.767	0.360	0.372	0.387
	SOLID++	2.648	3.925	6.148	0.447	0.580	0.659	0.735	0.981	1.262	4.156	4.780	5.835	0.376	0.378	0.397
	LEAF	2.136	3.588	5.920	0.487	0.613	0.698	0.792	0.997	1.286	4.133	4.812	5.787	0.368	0.371	0.385
	DSOF	1.864	3.538	5.846	0.438	0.596	0.703	0.748	0.984	1.305	4.186	4.894	5.973	0.364	0.388	0.414
	PROCEED	1.735	3.114	5.555	0.424	0.577	0.660	0.724	0.973	1.261	3.958	4.604	5.635	0.335	0.357	0.376
	Our	**1.582**	**2.965**	**5.284**	**0.407**	**0.556**	**0.648**	**0.702**	**0.954**	**1.243**	**3.927**	**4.464**	**5.503**	**0.324**	**0.341**	**0.364**
iTransformer	\	3.605	4.992	7.114	0.603	0.807	0.832	0.942	1.211	1.461	4.082	4.874	6.054	0.354	0.372	0.393
	OneNet	2.985	4.015	6.451	0.609	0.746	0.766	0.803	1.089	1.418	4.077	4.906	6.100	0.349	0.371	0.395
	SOLID++	2.804	4.278	6.582	0.455	0.601	0.688	0.875	1.139	1.403	4.070	4.816	6.024	0.342	0.365	0.384
	LEAF	2.763	4.044	6.631	0.512	0.652	0.726	0.867	1.107	1.395	4.155	4.893	5.974	0.357	0.379	0.391
	DSOF	2.354	3.857	**6.048**	0.494	0.663	0.806	0.837	1.079	1.342	4.136	4.862	6.025	0.351	0.372	0.384
	PROCEED	2.387	3.969	6.291	0.426	0.561	0.642	**0.742**	1.015	1.294	3.899	4.647	5.710	0.330	0.353	0.373
	Our	**2.245**	**3.826**	6.065	**0.405**	**0.537**	**0.626**	0.744	**0.975**	**1.144**	**3.685**	**4.458**	**5.532**	**0.317**	**0.343**	**0.360**

Table 1: MSE comparison of online forecasting methods across five benchmarks and three backbone architectures (TCN, PatchTST, iTransformer). The best results are marked in **bold**. The row marked \ denotes the backbone without any online adaptation.

Model	Method	ETTh2	ETTm1	Weather	ECL	Traffic	$\bar{\Delta}_{\text{MSE}}$
TCN	w/o PB	4.242	0.601	0.921	7.555	0.472	4.4%
	w/o TDC	4.404	0.625	0.965	7.915	0.490	8.8%
	w/o GOU	4.161	0.590	0.903	7.412	0.463	2.4%
	PADRE	**4.040**	**0.578**	**0.885**	**7.196**	**0.454**	-
PatchTST	w/o PB	3.408	0.558	0.995	4.817	0.353	3.6%
	w/o TDC	3.539	0.580	1.053	5.048	0.367	8.2%
	w/o GOU	3.343	0.548	0.986	4.770	0.350	2.2%
	PADRE	**3.277**	**0.537**	**0.966**	**4.631**	**0.343**	-
iTrans	w/o PB	4.248	0.549	0.993	4.786	0.354	4.6%
	w/o TDC	4.450	0.570	1.050	5.014	0.367	9.4%
	w/o GOU	4.167	0.538	0.983	4.695	0.347	2.8%
	PADRE	**4.045**	**0.523**	**0.954**	**4.558**	**0.340**	-

Table 2: Ablation study results. We report the average MSE with horizons in $\{24, 48, 96\}$ and the relative MSE increase ($\bar{\Delta}_{\text{MSE}}$) of the variants against PADRE on each dataset.

Online Protocol with Delayed Feedback. We strictly adhere to the practical online forecasting protocol defined in proceed [Zhao and Shen, 2025]. At time step t, the model receives the lookback window $\mathrm{X}_t \in \mathbb{R}^{L\times N}$ and predicts the future horizon $\mathrm{Y}_t \in \mathbb{R}^{H\times N}$. Crucially, we enforce a realistic constraint where the ground truth Y_t is not immediately available. The model can only be updated using historical samples $(\mathrm{X}_{t-H}, \mathrm{Y}_{t-H})$, where the ground truth has been fully observed. This introduces a feedback delay of H steps, creating a temporal gap that exacerbates the concept drift challenge.

Model Configuration. We evaluate performance across forecast horizons $H \in 24, 48, 96$. The lookback window length L is set based on the backbone architecture: L=60 for TCN-based backbone [Bai *et al.*, 2018] and L=512 for Transformer-based backbones (e.g., PatchTST [Nie *et al.*, 2023], iTransformer [Liu *et al.*, 2024]), consistent with Proceed. The number of regime clusters K is selected on the validation set by grid search over $K \in \{2, 4, 6, 8\}$. In practice, $K = 4$ works well across most datasets, balancing expressiveness (enough prototypes to capture diverse regimes) against over-fitting. M_{mem}

Implementation details. We utilize the Adam optimizer [Kingma and Ba, 2015]. All experiments are implemented in PyTorch on NVIDIA RTX 3090 GPUs. To account for randomness, we report the mean performance across five runs with different random seeds. The evaluation metric is Mean Squared Error (MSE).

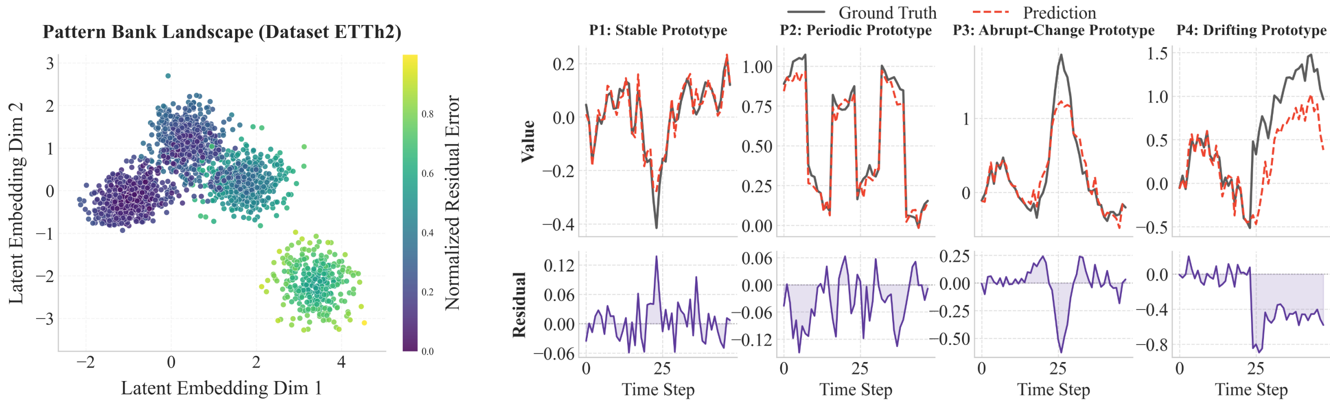
4.2 Generalization Performance

Consistent improvement across backbones. Table 1 reports MSE across all settings. Under TCN, PADRE improves over the strongest baseline (typically Proceed) by 5.3–8.7% across datasets. Similar gains appear for PatchTST (e.g., ETTh2: 1.735→1.582 at H=24; Traffic: 0.376→0.364 at H=96) and iTransformer (e.g., Weather: 11.1% and 9.6% reduction at H=24/48).

Horizon-agnostic gains. Relative improvements remain stable across forecast lengths rather than concentrating at particular horizons. Comparing H=24 and H=96, MSE reductions are consistently in the 6–15% range, indicating that PADRE's inductive bias generalizes across temporal scales. The largest gains occur on ECL and Traffic, where concept drift is heterogeneous, confirming that pattern aware adaptation is most beneficial when drift patterns are diverse.

4.3 Ablation Study

Table 2 evaluates three variants: w/o PB (no Pattern Bank; policy receives only local drift evidence), w/o TDC (no Tri-View Drift Construction; single-view residual encoding), and w/o GOU (no Gated Online Update; fixed learning rate). Removing TDC causes the largest degradation: average MSE increases by 8.8%, 8.2%, and 9.4% on TCN, PatchTST, and iTransformer respectively, with pronounced drops on ECL and Weather. This confirms that enforcing shared geometry across online, delayed, and memory views is central to extracting reliable adaptation signals. Eliminating PB leads to



(a) Latent Landscape

(b) Representative Prototypes & Residuals

Figure 3: Visualization of the Pattern Bank on ETTh2. **(a)** The learned latent embedding space shows four distinct regimes. Each point corresponds to a time series window, colored by its normalized residual error (darker colors indicate lower error). **(b)** Representative Prototypes & Residuals, showing ground truth (black), forecasting (red dashed), and residuals (purple, bottom). The model effectively distinguishes between stable, periodic, and drifting behaviors.

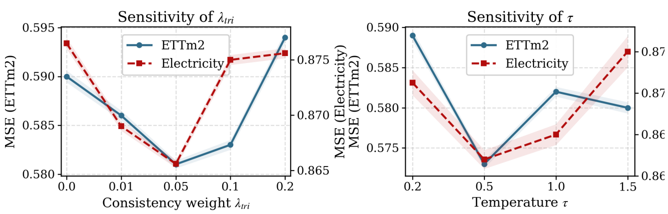


Figure 4: Sensitivity analysis of key hyper-parameters on ETTm2 and Electricity datasets.

moderate but consistent loss (3.6–4.6% increases), especially on Traffic and ECL, suggesting that reusing regime–residual prototypes helps recognize recurring drift patterns in complex, multi-seasonal series. Removing GOU produces 2–3% degradation; the model adapts using geometric cues but becomes less selective between transient shocks and persistent shifts. Pattern Bank brings sample efficiency by amortizing past drift experience; gating refines these signals into stable, drift-aware update magnitudes.

4.4 Case Study

In this section, we project the regime residual embeddings of ETTh2 dataset. Figure 3 probes the semantics of the Pattern Bank on ETTh2 by visualizing the regime–residual embedding, i.e., the concatenation of a regime representation and its regime-conditioned residual prototype. In Figure 3(a), the 2D projection exhibits four separated clusters; points are colored by normalized residual error (darker is lower), and the clusters show systematically different error levels, indicating recurring regime-specific difficulty rather than uniformly distributed delayed-feedback episodes.

Figure 3(b) anchors these clusters with representative prototypes. Stable and well-tracked periodic regimes yield small, largely structureless residuals, whereas a single-peak regime produces localized residual bursts, and a drifting-trend regime exhibits accumulating bias, reflecting persistent mismatch. This supports our intended role of the bank: prototypes act as regime-conditioned "error labels" that summarize how the backbone typically fails under each regime. Online, PADRE retrieves a soft pattern prompt p_t from the bank and combines it with drift evidence and geometric consistency A_t to produce the update gate γ_t, enabling stronger updates for coherent, systematic shifts and conservative updates for transient or poorly explained events.

4.5 Hyperparameter Analysis

We further examine PADRE's sensitivity to the geometric consistency weight λ_{tri} and the Pattern-Bank temperature τ (Fig.4). Performance is stable across a broad range, while extreme settings degrade results: too small λ_{tri} yields noisy, weakly coupled drift views, whereas overly large λ_{tri} over-regularizes the tri-view space. A similar unimodal pattern appears when sweeping the temperature τ: overly sharp attention (τ=0.2) and overly flat attention (τ=1.0) both underperform the default τ=0.5.

5 Conclusion

In this paper, we highlight the delayed residuals should not be used as raw supervision, but as noisy observations of latent drift states, and instantiated the idea with a Tri-View drift representation that aggregates online context, backward residual trajectories, and episodic drift memory. On top of this space, PADRE combines an offline Pattern Bank of regime residual prototypes with a prompt guided update policy, which outputs horizon aware update gates to modulate gradient steps. Experiments across diverse benchmarks, horizons, and backbone forecasters show that this design yields consistent and often substantial improvements over recent OTSF baselines, while ablations highlight the complementary roles of Tri-View geometry and pattern-bank guidance. A natural next step is utility and budget aware adaptation: learning update gates under constraints on computation, update frequency, or risk-sensitive costs, turning γ_t into a principled resource allocation mechanism rather than a purely accuracy driven scalar.

Acknowledgments

This work is supported by the grants from the National Natural Science Foundation of China (No.62272398), Sichuan Science and Technology Program (No.2024NSFJQ0019) and Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM211).

References

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

[Brenner *et al.*, 2025] Manuel Brenner, Elias Weber, Georgia Koppe, and Daniel Durstewitz. Learning interpretable hierarchical dynamical systems models from time series data. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28*, 2025.

[Cai *et al.*, 2025] Ruichu Cai, Haiqin Huang, Zhifan Jiang, Zijian Li, Changze Zhou, Yuequn Liu, Yuming Liu, and Zhifeng Hao. Disentangling long-short term state under unknown interventions for online time series forecasting. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 15641–15649. AAAI Press, 2025.

[Chen *et al.*, 2024] 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.

[Chen *et al.*, 2025] Weiqi Chen, Zhaoyang Zhu, Yifan Zhang, Lefei Shen, Linxiao Yang, Qingsong Wen, and Liang Sun. Learning to extrapolate and adjust: Two-stage meta-learning for concept drift in online time series forecasting. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025, Montreal, Canada, August 16-22, 2025*, pages 4869–4877. ijcai.org, 2025.

[Dama *et al.*, 2025] Srinath Dama, Kevin Course, and Prasanth B. Nair. Shifting time: Time-series forecasting with khatri-rao neural operators. In *Forty-second International Conference on Machine Learning*, 2025.

[Demirel and Holz, 2025] Berken Utku Demirel and Christian Holz. Shifting the paradigm: A diffeomorphism between time series data manifolds for achieving shift-invariancy in deep learning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28*, 2025.

[Dong *et al.*, 2024] Zheng Dong, Renhe Jiang, Haotian Gao, Hangchen Liu, Jinliang Deng, Qingsong Wen, and Xuan Song. Heterogeneity-informed meta-parameter learning for spatiotemporal time series forecasting. In Ricardo Baeza-Yates and Francesco Bonchi, editors, *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pages 631–641. ACM, 2024.

[Gupta *et al.*, 2025] Aditi Gupta, Adeiza James Onumanyi, Satyadev Ahlawat, Yamuna Prasad, and Virendra Singh. B-CAVE: A robust online time series change point detection algorithm based on the between-class average and variance evaluation approach. *IEEE Trans. Knowl. Data Eng.*, 37(1):75–88, 2025.

[He *et al.*, 2025] Hui He, Qi Zhang, Kun Yi, Kaize Shi, Zhendong Niu, and Longbing Cao. Distributional drift adaptation with temporal conditional variational autoencoder for multivariate time series forecasting. *IEEE Trans. Neural Networks Learn. Syst.*, 36(4):7287–7301, 2025.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[Lau *et al.*, 2025] Ying-yee Ava Lau, Zhiwen Shao, and Dit-Yan Yeung. Fast and slow streams for online time series forecasting without information leakage. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*, 2025.

[Lee *et al.*, 2025] Thomas L. Lee, William Toner, Rajkarn Singh, Artjom Joosen, and Martin Asenov. Lightweight online adaption for time series foundation model forecasts. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, 2025.

[Li *et al.*, 2025] Xingwang Li, Fei Teng, Minbo Ma, Zhen Qin, Jinhong Guo, Ji Xu, and Tianrui Li. Multi-level transfer learning for irregular clinical time series prediction. *Knowl. Based Syst.*, 324:113825, 2025.

[Li *et al.*, 2026a] Xingwang Li, Fei Teng, Tianrui Li, and Qiang Duan. Preference guided meta-learning for cross domain time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 38(4):2366–2379, 2026.

[Li *et al.*, 2026b] Xingwang Li, Fei Teng, Xin Wu, and Qiang Duan. Uncovering hidden degeneration: A physics-guided bidirectional inference framework for industrial time series prediction. In *Fortieth AAAI Conference on Artificial Intelligence, 2026, Singapore, January 20-27, 2026*, pages 23204–23211. AAAI Press, 2026.

[Liberti *et al.*, 2014] Leo Liberti, Carlile Lavor, Nelson Maculan, and Antonio Mucherino. Euclidean distance geometry and applications. *SIAM review*, 56(1):3–69, 2014.

[Liu *et al.*, 2024] 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 *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

[Liu *et al.*, 2025] Jiashuo Liu, Nabeel Seedat, Peng Cui, and Mihaela van der Schaar. Going beyond static: Understanding shifts with time-series attribution. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*, 2025.

[Na *et al.*, 2025] Jihye Na, Youngeun Nam, Junhyeok Kang, and Jae-Gil Lee. Mitigating source label dependency in time-series domain adaptation under label shifts. In Luiza Antonie, Jian Pei, Xiaohui Yu, Flavio Chierichetti, Hady W. Lauw, Yizhou Sun, and Srinivasan Parthasarathy, editors, *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, V.2, KDD 2025, Toronto ON, Canada, August 3-7, 2025*, pages 2150–2161. ACM, 2025.

[Nie *et al.*, 2023] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5*, 2023.

[Pham *et al.*, 2023] Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven C. H. Hoi. Learning fast and slow for online time series forecasting. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[Shao *et al.*, 2025] Zezhi Shao, Fei Wang, Yongjun Xu, Wei Wei, Chengqing Yu, Zhao Zhang, Di Yao, Tao Sun, Guangyin Jin, Xin Cao, Gao Cong, Christian S. Jensen, and Xueqi Cheng. Exploring progress in multivariate time series forecasting: Comprehensive benchmarking and heterogeneity analysis. *IEEE Trans. Knowl. Data Eng.*, 37(1):291–305, 2025.

[Wang *et al.*, 2025a] Jie Wang, Tianrui Li, Yan Yang, Shiqian Chen, and Wanming Zhai. Diagllm: multimodal reasoning with large language model for explainable bearing fault diagnosis. *Science China Information Sciences*, 68(6):160103, 2025.

[Wang *et al.*, 2025b] Yihang Wang, Yuying Qiu, Peng Chen, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series forecasting model with unified representation and adaptive transfer. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19*, 2025.

[Wen *et al.*, 2023] 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.

[Witulska *et al.*, 2026] Justyna Witulska, Marta Hendler, Magdalena Kasprowicz, Marek Czosnyka, Ireneusz Jablonski, and Agnieszka Wylomanska. Identifying the temporal distribution structure in multivariate data for time-series segmentation based on two-sample test. *Inf. Fusion*, 125:103445, 2026.

[Wu *et al.*, 2025] Xingjian Wu, Xiangfei Qiu, Zhengyu Li, Yihang Wang, Jilin Hu, Chenjuan Guo, Hui Xiong, and Bin Yang. CATCH: channel-aware multivariate time series anomaly detection via frequency patching. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28*, 2025.

[Wu *et al.*, 2026] Xin Wu, Fei Teng, Ji Zhang, Xingwang Li, and Yuxuan Liang. Eris : An energy-guided feature disentanglement framework for out-of-distribution time series classification. *Information Fusion*, 135:104407, 2026.

[Zhang *et al.*, 2025] Weiyang Zhang, Xinyang Chen, Yu Sun, Weili Guan, and Liqiang Nie. Batch training for streaming time series: A transferable augmentation framework to combat distribution shifts. *Trans. Mach. Learn. Res.*, 2025.

[Zhao and Shen, 2025] 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 V. 1*, pages 2020–2031, 2025.

[Zhao *et al.*, 2025] Zhilin Zhao, Longbing Cao, and Yuan-Yu Wan. Mixture of online and offline experts for non-stationary time series. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 22867–22874. AAAI Press, 2025.