PROGRESSIVE ADAPTATION WITH VERSION-AWARE EXPERTS FOR ONLINE TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Online time series forecasting (OTSF) requires continual adaptation to evolving data distributions, while supervision for multi-step forecasts becomes progressively available. Waiting for fully matured feedback delays adaptation, and historical expert credit can become stale as experts continue to evolve. We propose **PAVE**, a multi-expert framework that organizes online adaptation around *feedback maturity* and *capability validity*. As partial feedback arrives, PAVE rapidly adapts expert selection through a feedback-driven online routing correction and local responsibility supervision of a neural router. Once all targets of a forecasting record have matured, PAVE assigns sample-level responsibility from prediction-time expert outputs and validates this historical ownership against the experts' current capabilities. The resulting version-aware evidence determines whether a capability should be preserved, rebased, or recovered through responsibility-weighted subspace protection and targeted replay. Experiments on four multivariate forecasting benchmarks show that PAVE consistently outperforms strong online forecasting baselines, while ablation and diagnostic analyses support the effectiveness of progressive routing, version-aware credit assignment, and preservation–recovery.

1 INTRODUCTION

Online time series forecasting is increasingly important in streaming applications such as energy, finance, environmental monitoring, and industrial systems, where observations arrive continuously and the underlying data-generating processes may evolve over time (Melgar-García et al., 2026). Time series forecasters deployed in evolving environments must adapt as distribution shifts erode models trained on historical data (Gama et al., 2014; Kim et al., 2022; Liu et al., 2022). In multi-step online forecasting, however, supervision is also delayed and progressive: a forecast issued at origin o contains multiple future targets that become observable sequentially before the whole forecasting record is fully matured (Zhao & Shen, 2025; Lau et al., 2025; Kim et al., 2025). Waiting for complete feedback sacrifices timeliness, whereas allowing sparse early observations to determine long-term expert ownership risks premature credit assignment. We therefore adopt a simple principle: *evidence of different maturity should support different adaptation roles*. Mixture-of-experts architectures provide a natural substrate for such role separation (Shazeer et al., 2017; Fedus et al., 2022): partial feedback can support fast routing adaptation, whereas only fully matured evidence should determine durable expert-level ownership.

Introducing mixture of experts raises a following challenge, because the experts themselves continue to evolve. We use *expert version* to denote the state of the same continuously updated expert at a particular online time, including its parameters and induced predictive capability. Once a forecasting record is fully observed, stored prediction-time expert outputs reveal which experts historically accounted for that record. However, subsequent online updates may move the same expert to a different version, making this historical ownership stale. Blindly preserving historically successful evidence can restrict beneficial adaptation, whereas ignoring it can lead to forgetting of reusable capabilities, a central concern in continual and online learning (Pham et al., 2023; Saha et al., 2021; Farajtabar et al., 2020). We therefore distinguish *historical ownership* from *current capability validity*. Historical credit computed from prediction-time outputs records which expert earned responsibility for the issued forecast, while version validation asks whether that capability remains valid under the expert's current version. Long-term evidence should affect an evolving expert only after this distinction is resolved.

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.

Based on these observations, we propose **Progressive Adaptation with Version-aware Experts (PAVE)**, centered on an *evidence lifecycle* that couples feedback maturity with the validity of evolving expert capabilities. PAVE uses progressively revealed partial feedback for fast routing adaptation, while only fully matured records establish sample-level historical responsibility for long-term expert adaptation. Crucially, such historical ownership is not assumed to remain valid as experts continue to evolve. PAVE therefore re-evaluates the current version of each responsible expert against the capability associated with its prediction-time version and determines whether the associated capability should be preserved, rebased after beneficial or neutral evolution, or recovered after harmful degradation. This separation allows PAVE to exploit early feedback without letting incomplete or stale evidence directly govern long-term expert adaptation.

Our main contributions are summarized as follows:

- We identify a key mismatch in multi-step online forecasting: partial feedback is timely but unreliable for long-term expert credit, whereas fully matured historical credit can become stale as continuously updated experts evolve. We propose **PAVE**, a progressive evidence-lifecycle framework that jointly accounts for feedback maturity and capability validity.
- Technically, PAVE combines **progressive routing** from partial feedback with **version-aware expert adaptation** from matured feedback: it assigns historical responsibility from prediction-time outputs, revalidates that ownership by comparing the prediction-time and current versions of the responsible expert, and selectively preserves, rebases, or recovers the associated capability.
- Experiments on four real-world multivariate benchmarks show that PAVE achieves the best or tied-best MSE in all 12 dataset–horizon settings, with an average 5.7% reduction over the strongest competing baseline. Ablations, matched controls, and mechanism-level diagnostics further validate the proposed evidence lifecycle, especially at long forecasting horizons.

2 Related Work

OTSF methods address non-stationarity through rapid adaptation, drift-aware updates, dynamic experts, or progressively available supervision (Pham et al., 2023; Zhang et al., 2023; Zhao & Shen, 2025; Lau et al., 2025; Hong et al., 2026). PAVE instead assigns distinct roles to feedback maturity: partial feedback adapts routing, while fully matured feedback establishes sample-level ownership that is reassessed as the responsible expert evolves. This differs from simply exposing every model component to targets as soon as they arrive, and from combining experts without tracking whether their assigned capabilities remain valid.

Continual-learning methods use replay or gradient subspaces to reduce interference with prior knowledge (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019; 2020; Zeng et al., 2019). PAVE likewise protects useful capability, but does not treat historical knowledge as permanently valid: responsibility is weighted and validated against current capability before deciding to preserve, rebase, or recover. Stable evidence supports expert-specific predictive subspaces, whereas evidence of harmful degradation is used for targeted replay instead of permanent preservation. Extended positioning is provided in Appendix A.

3 Online Forecasting with Progressive Feedback

Consider a multivariate stream $\{\mathbf{y}_t\}_{t=1}^{T}$ with $\mathbf{y}_t \in \mathbb{R}^C$. At each rolling origin t, the model observes $\mathbf{X}_t = [\mathbf{y}_{t-L+1}, \ldots, \mathbf{y}_t] \in \mathbb{R}^{L\times C}$ and predicts $\mathbf{Y}_t = [\mathbf{y}_{t+1}, \ldots, \mathbf{y}_{t+H}] \in \mathbb{R}^{H\times C}$. We use E continuously adapting experts, $\hat{\mathbf{Y}}_t^{(e)} = f_e(\mathbf{X}_t; \theta_e^t)$. For horizon h and channel c, routing weights $\mathbf{p}_{t,h,c} \in \Delta_E$ give

$$\hat{y}_{t,h,c} = \sum_{e=1}^{E} p_{t,h,c,e}\hat{y}_{t,h,c}^{(e)}. \tag{1}$$

Once the full target window is observed, the origin-level loss is $\mathcal{L}_t = \frac{1}{HC}\sum_{h,c}(\hat{y}_{t,h,c} - y_{t+h,c})^2$.

Multi-step supervision is delayed and progressive: the h-th target of a forecast issued at origin o becomes observable at time $o + h$. Hence the feedback released at step t is

$$\mathcal{F}_t = \{(o, h) \mid o + h = t,\ 1 \le h \le H\}. \tag{2}$$

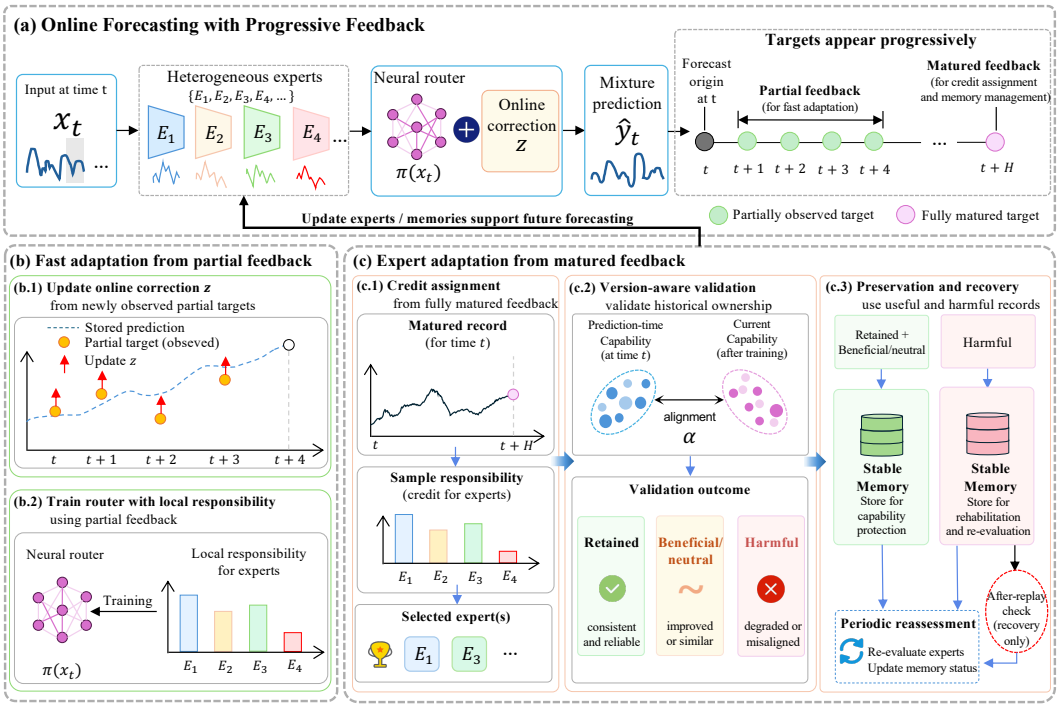


Figure 1: **PAVE overview.** Partial feedback adapts routing; fully matured feedback assigns responsibility, which version validation directs to Stable Memory or Recovery Memory.

We follow a leakage-free rolling protocol: at each step, the model observes $\mathbf{y}_t$, releases $\mathcal{F}_t$, adapts using only causally available information, and then predicts $\mathbf{y}_{t+1:t+H}$. Thus each record progresses from partial horizon-level evidence to fully matured supervision without using any target before it is observed.

Delayed-routing guarantee. The progressive router admits an idealized delayed online-learning view. For one horizon–channel coordinate, let $f_t(\mathbf{p}) = (\sum_e p_e \hat{y}_{t,e} - y_t)^2$ and $R_T(\mathbf{u}) = \sum_t [f_t(\mathbf{p}_t) - f_t(\mathbf{u})]$. Let $\mathbf{g}_t = \nabla f_t(\mathbf{p}_t)$ with $\|\mathbf{g}_t\|_\infty \le G$, D_T denote cumulative feedback delay, and $V_T(\eta) = \sum_t \|\mathbf{g}_t - \mathbf{m}_t(\eta)\|_\infty^2$ denote the mismatch between the realized routing gradient and the neural-prior signal.

Theorem 1 (Delayed routing regret)**.** *For the idealized dense router with fixed expert predictions and without decay, clipping, or top-k truncation, for any fixed* $\mathbf{u} \in \Delta_E$,

$$R_T(\mathbf{u}) \le \frac{\log E}{\eta} + \frac{\eta}{2} V_T(\eta) + 2\eta G^2 D_T. \tag{3}$$

The bound separates neural-prior mismatch from delayed-feedback cost; progressive feedback reduces the latter relative to waiting for full record maturation. The definition of $\mathbf{m}_t(\eta)$, complete derivation, proof, and corollaries are given in Appendix C.

4 Progressive Evidence Adaptation with Version-Aware Experts

Figure 1 summarizes PAVE as a progressive evidence lifecycle connecting fast routing, version-aware credit assignment, and expert preservation or recovery. Partial feedback is used for timely routing adaptation, whereas fully matured feedback establishes historical expert responsibility. Because each expert may have changed since the forecast was issued, PAVE treats this responsibility as tied to the expert's prediction-time version and validates it against the expert's current version before using it for long-term adaptation.

4.1 FAST ROUTING FROM PROGRESSIVE EVIDENCE

Given the current context $\mathbf{X}_t$, a horizon–channel neural router produces a dense contextual prior $\boldsymbol{\pi}_{t,h,c} \in \Delta^E$ over experts. To rapidly incorporate newly observed outcomes, PAVE maintains an online correction $\mathbf{z} \in \mathbb{R}^{H\times C\times E}$ in addition to the learned router. The effective routing distribution is

$$\mathbf{p}_{t,h,c} = \mathrm{softmax}\left(\log \boldsymbol{\pi}_{t,h,c} + \mathbf{z}_{h,c}\right). \tag{4}$$

Here, $\boldsymbol{\pi}$ captures context-dependent routing preference learned across the stream, while $\mathbf{z}$ provides a lightweight correction driven directly by recent feedback. Optional top-k sparsification is applied after the correction.

Feedback-driven correction. Consider a forecast issued at origin o, whose h-th target becomes observable at the current step $t = o + h$. Let $\hat{y}^{(e)}_{o,h,c}$ and $\hat{y}_{o,h,c}$ denote the stored prediction-time output of expert e and the corresponding mixture prediction. The newly observed target defines the centered mixture-weight gradient

$$g^{\mathrm{corr}}_{o,h,c,e} = 2\left(\hat{y}_{o,h,c} - y_{t,c}\right)\left(\hat{y}^{(e)}_{o,h,c} - \hat{y}_{o,h,c}\right), \qquad z_{h,c,e} \leftarrow z_{h,c,e} - \eta_z g^{\mathrm{corr}}_{o,h,c,e}. \tag{5}$$

The ordinary mixture-weight gradient is $2(\hat{y}_{o,h,c} - y_{t,c})\hat{y}^{(e)}_{o,h,c}$; equation 5 subtracts a common term from all expert coordinates, which leaves the normalized exponentiated-gradient update unchanged. Thus, z acts as an online log-weight correction rather than being updated by the Euclidean gradient with respect to the softmax logits. The correction signal uses stored prediction-time expert outputs and the realized target, without re-evaluating experts after they have changed, and therefore assigns causal feedback to the forecast actually issued.

Local responsibility from partial feedback. The same newly observed target also supervises the neural router through a separate credit-assignment path. Using the stored prediction-time expert outputs, we define the local expert error and responsibility as

$$\ell^{\mathrm{loc}}_{o,h,c,e} = \left(\hat{y}^{(e)}_{o,h,c} - y_{t,c}\right)^2, \qquad \mathbf{r}^{\mathrm{loc}}_{o,h,c} = \mathrm{softmax}\left(-\frac{\boldsymbol{\ell}^{\mathrm{loc}}_{o,h,c}}{\tau_{\mathrm{loc}}}\right), \tag{6}$$

and measure the certainty of this local assignment by

$$\kappa^{\mathrm{loc}}_{o,h,c} = 1 - \frac{\mathcal{H}\left(\mathbf{r}^{\mathrm{loc}}_{o,h,c}\right)}{\log E}. \tag{7}$$

The confidence is small when responsibility is nearly uniform and approaches one when the partial feedback clearly favors a particular expert. For all newly observed targets, the current neural router is re-evaluated on the stored context X_o, yielding $\boldsymbol{\pi}^{\mathrm{cur}}_{o,h,c}$, and is updated using the partial mixture loss together with confidence-weighted local credit,

$$\mathcal{L}^{\mathrm{partial}}_{\mathrm{router}} = \mathcal{L}^{\mathrm{partial}}_{\mathrm{mix}} + \lambda_{\mathrm{loc}}\mathcal{L}_{\mathrm{credit}} - \lambda_{\mathrm{ent}}\mathcal{H}_{\mathrm{router}}, \tag{8}$$

where $\mathcal{L}_{\mathrm{credit}} = \frac{1}{|\mathcal{F}_t|C}\sum_{(o,h)\in\mathcal{F}_t}\sum_{c=1}^{C}\kappa^{\mathrm{loc}}_{o,h,c}\,\mathrm{CE}\left(\mathbf{r}^{\mathrm{loc}}_{o,h,c} \,\|\, \boldsymbol{\pi}^{\mathrm{cur}}_{o,h,c}\right)$. $\mathcal{L}^{\mathrm{partial}}_{\mathrm{mix}}$ uses the corresponding effective routing weights with the stored prediction-time expert outputs, and $\mathcal{H}_{\mathrm{router}}$ is the mean entropy of the current dense router priors. The correction state and stored expert outputs are treated as constants in this router update. Thus, partial feedback drives two complementary forms of fast adaptation: $\mathbf{z}$ directly corrects routing from recent forecasting errors, while $\mathbf{r}^{\mathrm{loc}}$ provides confidence-weighted supervision for the current neural router. Neither signal is used for long-term expert preservation, which is determined only after the forecasting record becomes fully matured.

4.2 VERSION-AWARE EXPERT RESPONSIBILITY

Once all H targets of origin o have been observed, the record becomes fully matured and provides fully matured feedback. PAVE then evaluates the stored prediction-time outputs over the complete forecasting window and assigns sample-level expert responsibility:

$$\bar{\ell}_{o,e} = \frac{1}{HC}\sum_{h=1}^{H}\sum_{c=1}^{C}\left(\hat{y}^{(e)}_{o,h,c} - y_{o+h,c}\right)^2, \qquad \mathbf{r}_o = \mathrm{softmax}\left(-\frac{\bar{\boldsymbol{\ell}}_o}{\tau_s}\right), \qquad \kappa_o = 1 - \frac{\mathcal{H}(\mathbf{r}_o)}{\log E}. \tag{9}$$

We initialize the performance reference as $\bar{\ell}^{\mathrm{ref}}_{o,e} = \bar{\ell}_{o,e}$. The top-$K_c$ experts whose responsibility exceeds a minimum threshold are retained for validation. Unlike the routing distribution $\mathbf{p}$, $\mathbf{r}_o$ represents historical ownership revealed by fully matured feedback and is tied to the expert's prediction-time version. Computing it from stored outputs prevents later updates from retroactively changing which expert earned credit.

We initialize the performance reference as $\bar{\ell}^{\mathrm{ref}}_{o,e} = \bar{\ell}_{o,e}$. The top-$K_c$ experts whose responsibility exceeds a minimum threshold are retained for validation. Unlike the routing distribution $\mathbf{p}$, $\mathbf{r}_o$ represents historical ownership revealed by fully matured feedback and is tied to the expert's prediction-time version. Computing it from stored outputs prevents later updates from retroactively changing which version earned credit.

Capability validity across expert versions. An expert version denotes the state of the same expert at a particular online time. PAVE compares the prediction-time and current versions through a lightweight capability sketch rather than storing full historical experts. For each expert, a fixed Gaussian projection $(320 \to 32)$ is applied to its pooled encoder representation and ℓ_2-normalized. We store $\mathbf{s}^{\mathrm{pred}}_{o,e}$ at prediction time and recompute $\mathbf{s}^{\mathrm{cur}}_{o,e}$ on the same context with the current expert. A reference sketch $\mathbf{s}^{\mathrm{ref}}_{o,e}$ is initialized as $\mathbf{s}^{\mathrm{pred}}_{o,e}$, and their alignment is

$$a_{o,e} = \frac{1 + \cos\left(\mathbf{s}^{\mathrm{ref}}_{o,e}, \mathbf{s}^{\mathrm{cur}}_{o,e}\right)}{2}. \tag{10}$$

Thus, $r_{o,e}$ measures historical ownership, whereas $a_{o,e}$ measures its validity under the current expert version.

A low alignment does not necessarily imply forgetting, since an expert may change while maintaining or improving its forecasting quality. Let $\bar{\ell}^{\mathrm{cur}}_{o,e}$ denote the loss of the current expert on the same matured record. PAVE classifies each responsible expert as

$$\mathcal{V}_{o,e} = \begin{cases} \text{retained}, & a_{o,e} \geq \tau_a, \\ \text{beneficial/neutral}, & a_{o,e} < \tau_a \ \wedge \ \bar{\ell}^{\mathrm{cur}}_{o,e} \leq (1+\delta)\bar{\ell}^{\mathrm{ref}}_{o,e} + \epsilon, \\ \text{harmful}, & a_{o,e} < \tau_a \ \wedge \ \bar{\ell}^{\mathrm{cur}}_{o,e} > (1+\delta)\bar{\ell}^{\mathrm{ref}}_{o,e} + \epsilon, \end{cases} \tag{11}$$

where δ and ϵ denote relative and absolute tolerances for performance degradation, respectively. High alignment indicates that the current expert version remains close to the reference version, so the associated historical capability is treated as retained. Low alignment indicates substantial version change, but does not by itself imply forgetting: the new version may have preserved or improved predictive quality. We therefore use the current loss only to determine the direction of such substantial change. Beneficial or neutral evolution is *rebased* by replacing the historical reference with the current capability, i.e., $\mathbf{s}^{\mathrm{ref}}_{o,e} \leftarrow \mathbf{s}^{\mathrm{cur}}_{o,e}$ and $\bar{\ell}^{\mathrm{ref}}_{o,e} \leftarrow \bar{\ell}^{\mathrm{cur}}_{o,e}$, whereas harmful degradation is passed to recovery. Rebase is thus an operation that refreshes the reference, not a persistent memory state. This asymmetry makes alignment the primary validity signal and invokes loss direction only after substantial version change. Although high alignment is not a formal guarantee of unchanged predictive performance, we further examine the potential high-alignment failure mode in Appendix E by conditioning on historical owners with $a_{o,e} \geq \tau_a$. Among these cases, 95.8% show no performance degradation, while only 1.8% and 0.84% exhibit relative loss increases above 10% and 20%, respectively.

4.3 Evidence-conditioned preservation and recovery

PAVE maintains expert-specific Stable and Recovery memories according to the version-aware decision. Their evidence scores are $q^{\mathrm{sta}}_{o,e} = \kappa_o r_{o,e} a_{o,e}$, and $q^{\mathrm{rec}}_{o,e} = \kappa_o r_{o,e}(1 - a_{o,e})$. The former emphasizes confidently assigned capabilities that remain valid, whereas the latter emphasizes confidently owned capabilities that have substantially changed. Beneficial or neutral samples are rebased and treated as stable evidence; harmful samples enter Recovery Memory.

Stable evidence for subspace preservation. Let $\mathbf{h}_{i,e} \in \mathbb{R}^d$ denote the representation presented to the prediction head of expert e for stable evidence i. We construct the responsibility-weighted uncentered second-moment matrix

$$\mathbf{M}_e = \frac{1}{m_e} \sum_{i \in \mathcal{S}_e} q^{\mathrm{sta}}_{i,e} \mathbf{h}_{i,e} \mathbf{h}^{\top}_{i,e}, \qquad m_e = \sum_{i \in \mathcal{S}_e} q^{\mathrm{sta}}_{i,e}. \tag{12}$$

The leading eigenvectors of $\mathbf{M}_e$ define an expert-specific predictive subspace $\mathbf{U}_e$ supported by reliable stable evidence. The weights prevent weakly owned samples from dominating protected directions, while expert-specific construction avoids imposing a shared stability geometry on heterogeneous experts.

PAVE updates the full expert online, while applying subspace protection only to its prediction head. Let $\mathbf{G}_e$ denote the gradient of the prediction-head weights. We decompose and modify it as

$$\mathbf{G}_e^{\parallel} = \mathbf{G}_e \mathbf{U}_e \mathbf{U}_e^{\top}, \qquad \mathbf{G}_e^{\perp} = \mathbf{G}_e - \mathbf{G}_e^{\parallel}, \qquad \widetilde{\mathbf{G}}_e = \mathbf{G}_e^{\perp} + \gamma_e \mathbf{G}_e^{\parallel}. \tag{13}$$

Rather than imposing fixed protection, PAVE conditions its strength on the amount of validated stable evidence:

$$\mu_e = \frac{m_e}{m_e + m_0}, \qquad \gamma_e = \frac{1}{1 + \eta_e \lambda_{\mathrm{sub}} \mu_e}, \tag{14}$$

where m_0 controls evidence-mass saturation, η_e is the expert learning rate, and λ_{sub} controls protection strength. As reliable evidence accumulates, μ_e increases and γ_e decreases, progressively strengthening protection along the corresponding predictive directions.

Targeted recovery. Subspace protection cannot restore a capability that has already degraded. PAVE therefore selectively replays evidence from Recovery Memory. For recovery record i assigned to expert e, we optimize $\mathcal{L}_{\mathrm{rec},i}^{(e)} = r_{i,e}\left[\mathcal{L}_{\mathrm{pred},i}^{(e)} + \lambda_{\mathrm{sk}} \left\| \mathbf{s}_{i,e}^{\mathrm{cur}} - \mathrm{sg}\left(\mathbf{s}_{i,e}^{\mathrm{ref}}\right)\right\|_2^2\right]$, where $\mathcal{L}_{\mathrm{pred},i}^{(e)}$ is the forecasting loss, λ_{sk} controls sketch distillation, and $\mathrm{sg}(\cdot)$ denotes stop-gradient. The reference sketch $\mathbf{s}_{i,e}^{\mathrm{ref}}$ is initialized from the prediction-time sketch and updated to the current sketch after rebasing. The prediction term restores forecasting quality, while sketch distillation encourages recovery of the associated historical capability. Recovery updates are subjected to the same stable-subspace protection, reducing interference with capabilities that remain valid.

Both memories are periodically reassessed as experts evolve. Stable evidence is retained, rebased, or migrated to Recovery according to the same version-aware criterion. Recovery evidence is additionally reassessed after targeted replay: recovered items return to Stable Memory, while repeatedly unsuccessful items are discarded. Reassessment is necessary because both memories describe evidence relative to a changing expert version: currently valid evidence may later become stale, while recovered capability should no longer consume replay effort.

Overall online procedure. At each step, PAVE processes all causally available partial and matured feedback before issuing the current forecast, followed by periodic memory reassessment when triggered. Detailed pseudocode is provided in Appendix B (Algorithm 1).

5 Experiments

We evaluate PAVE from three perspectives: ❶ **Main Result.** Compare PAVE with baselines under widely used protocols; ❷ **Ablation Study.** Evaluate the contribution of PAVE's main components; ❸ **Mechanistic Analysis.** Examine whether its internal behavior supports the proposed progressive evidence lifecycle.

5.1 Experimental setup

Datasets and protocol. We use ETTh2, ETTm1, Weather (WTH), and Electricity (ECL), with input length $L = 60$ and horizons $H \in \{1, 24, 48\}$. Models are initialized on the historical training split and adapted sequentially over identical chronological test streams. Under the causal protocol of Section 3, a target is usable only after observation, so partial feedback arrives before a record is fully matured.

Baselines and metrics. We compare with DSOF (Lau et al., 2025), DynaME (Hong et al., 2026), ER (Rolnick et al., 2019), FSNet (Pham et al., 2023), OneNet (Zhang et al., 2023), PROCEED (Zhao & Shen, 2025), and PatchTST-DGrad (Nie et al., 2023). MSE is primary; per-setting MAE results are deferred to Appendix D.

Table 1: MSE (mean ± standard deviation over three runs; lower is better). Bold and underline mark the best and second-best results. Rankings are based on means rounded to three decimals; ties share rank.

Method	ETTh2 1	ETTh2 24	ETTh2 48	ETTm1 1	ETTm1 24	ETTm1 48
DSOF	0.754±0.044	3.552±0.443	5.625±0.388	0.114±0.001	0.600±0.006	0.769±0.010
DynaME	0.445 ± 0.065	2.431 ± 0.205	5.274±0.247	**0.085 ± 0.002**	0.608±0.010	0.767±0.010
ER	0.527±0.013	2.530±0.244	5.041±0.249	0.091±0.002	0.590±0.013	0.811±0.012
FSNet	0.581±0.016	3.007±0.206	4.970 ± 0.143	0.138±0.012	1.022±0.038	1.416±0.067
OneNet	0.559±0.014	3.442±0.431	6.337±0.688	0.119±0.004	0.889±0.002	1.195±0.016
PROCEED	0.676±0.016	5.015±0.454	17.438±0.831	0.087 ± 0.002	0.568 ± 0.010	0.761 ± 0.010
PatchTST-DGrad	0.616±0.010	3.346±0.326	5.776±0.292	0.125±0.004	0.700±0.013	0.942±0.010
PAVE	**0.355 ± 0.005**	**2.408 ± 0.008**	**4.919 ± 0.074**	**0.085 ± 0.001**	**0.557 ± 0.016**	**0.759 ± 0.004**
Method	WTH 1	WTH 24	WTH 48	ECL 1	ECL 24	ECL 48
DSOF	0.209±0.000	0.534±0.003	0.734±0.006	2.561±0.052	7.594 ± 0.122	9.310±0.105
DynaME	0.186±0.001	0.436 ± 0.003	0.618 ± 0.005	3.465±0.078	14.804±0.205	17.415±0.211
ER	0.171±0.002	0.455±0.002	0.633±0.006	2.371 ± 0.062	7.784±0.169	9.279 ± 0.087
FSNet	0.238±0.003	0.608±0.003	0.787±0.008	6.131±0.173	17.618±0.405	20.290±0.222
OneNet	0.177±0.001	1.362±1.261	1.838±1.267	4.537±0.283	10.616±0.175	12.596±0.707
PROCEED	0.158 ± 0.003	0.452±0.001	0.652±0.005	6.895±0.194	18.681±0.209	22.406±0.232
PatchTST-DGrad	0.169±0.001	0.503±0.003	0.695±0.007	2.678±0.076	7.809±0.127	10.081±0.108
PAVE	**0.156 ± 0.000**	**0.425 ± 0.001**	**0.509 ± 0.003**	**2.301 ± 0.045**	**6.996 ± 0.075**	**8.207 ± 0.096**

Implementation details. PAVE uses four experts, a horizon–channel router with routing top-$K_r = 4$ and credit top-$K_c = 1$, correction learning rate 0.1, sketch dimension 32, responsibility threshold 0.3, alignment threshold 0.8, per-expert memory capacity 32, and subspace rank 16. All PAVE variants share initialization and offline training; optimizer, recovery, reassessment, and dataset-specific settings are in Appendix B.

5.2 Overall forecasting performance

Table 1 compares PAVE on four multivariate benchmarks with $H \in \{1, 24, 48\}$; results are mean ± standard deviation over three runs. We further test whether its gains arise simply from earlier partial feedback.

Table 2: Progressive-feedback control on three representative settings with matched causal horizon-level feedback access. Entries are MSE (mean ± standard deviation over three runs).

Method	ETTm1-24	WTH-48	ECL-48
DSOF + PF	0.571 ± 0.009	0.590 ± 0.006	8.779 ± 0.249
DynaME + PF	0.598 ± 0.011	0.519 ± 0.004	15.289 ± 0.217
ER + PF	0.582 ± 0.008	0.613 ± 0.003	8.677 ± 0.161
PAVE	**0.557 ± 0.016**	**0.509 ± 0.003**	**8.207 ± 0.096**

Overall forecasting accuracy. PAVE achieves the best or tied-best result in all 12 dataset–horizon settings, including 11 outright wins and one tie. Relative to the strongest competing baseline in each setting, PAVE reduces MSE by approximately 5.7% on average. Across the four datasets, the median relative improvement over the strongest baseline is 2.1%, 2.2%, and 6.3% for $H = 1$, 24, and 48, respectively, showing that the advantage is most pronounced at long horizons.

Controlling for progressive-feedback access. We further examine whether PAVE's gains can be attributed simply to earlier access to supervision. As shown in Table 2, on three representative settings, ETTm1-24, WTH-48, and ECL-48, we provide DSOF, DynaME, and ER with the same causally available horizon-level feedback as PAVE while retaining their respective online adaptation mechanisms.

Across the three representative settings examined, PAVE remains best-performing after matching causal progressive-feedback access. Compared with the strongest feedback-matched baseline in each setting, PAVE reduces MSE by 2.5% on ETTm1-24, 1.9% on WTH-48, and 5.4% on ECL-48. Thus, within these controlled settings, providing earlier partial feedback alone does not eliminate

PAVE's advantage, suggesting that how feedback of different maturity is used also contributes to the observed gains.

5.3 Ablation study

Figure 2 reports the relative MSE increase of each ablation over full PAVE, averaged equally across datasets at each horizon; complete per-setting results are provided in Appendix D. Operationally, *w/o Online Correction* disables $\mathbf{z}$, while *w/o Local Credit* sets $\lambda_{\text{loc}} = 0$ but retains the partial mixture loss. *w/o Version Awareness* treats effective alignment as one, so matured ownership is used without version-based reassessment; *w/o Direction Awareness* retains alignment-based detection but removes loss-direction gating and rebasing, making all low-alignment cases eligible for Recovery. *w/o Subspace Protection* applies raw expert gradients, *Unweighted Subspace* uses uniform Stable-sample weights only when estimating the protected subspace, and *w/o Recovery* disables the Recovery memory and replay branch. Full operational definitions are given in Appendix B.8.

Progressive routing. Both partial-feedback routing mechanisms contribute across horizons, with larger effects at longer horizons. At $H = 48$, removing the online correction or local responsibility increases MSE by 12.8% and 10.4%, respectively. The two components serve complementary roles: the correction $\mathbf{z}$ reacts directly to newly observed forecasting errors, whereas local responsibility provides confidence-weighted supervision to the neural router. Their larger long-horizon effects indicate the value of exploiting partial feedback before the full forecasting record matures.



Figure 2: Relative MSE increase (%) over full PAVE, averaged across datasets for each horizon. Larger values indicate stronger degradation.

Version-aware expert responsibility. Version-aware validation exhibits the strongest horizon-dependent effect. At $H = 48$, removing version awareness produces the largest degradation, 16.4%, while removing direction awareness increases MSE by 11.0%. The former directly reuses matured historical ownership without checking whether the responsible expert has changed, whereas the latter still detects version change but no longer distinguishes harmful degradation from beneficial or neutral evolution. These results support both revalidating historical credit under the current expert version and using predictive direction to decide between recovery and rebasing.

Preservation and recovery. Preservation and recovery show a similar long-horizon pattern. At $H = 48$, removing Stable-subspace protection or the Recovery branch increases MSE by 11.2% and 10.9%, respectively. Replacing responsibility-weighted Stable evidence with uniform weights when estimating the protected subspace still causes a 6.9% increase, while leaving the remaining lifecycle unchanged. This suggests that both the protected geometry and targeted restoration of degraded capabilities contribute to performance.

Overall, ablation effects are modest at short horizons but substantially larger at $H = 48$. With slower feedback maturation, more expert updates can occur before historical ownership is reused, increasing the importance of progressive routing, version validation, and preservation–recovery.

5.4 Understanding PAVE

We further examine three mechanisms underlying PAVE: whether historical expert credit becomes stale as experts evolve, whether targeted replay restores degraded capabilities, and how credit staleness varies with the forecasting horizon. All diagnostics use fully matured records and prediction-time references, so historical ownership is evaluated without being recomputed from the current expert outputs.

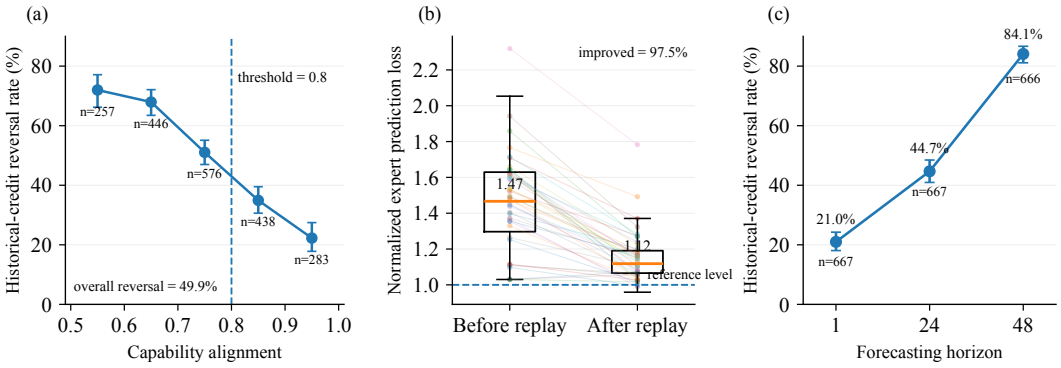


Figure 3: **Mechanistic analysis.** (a) Historical-credit reversal vs. capability alignment (95% Wilson CIs; dashed: $\tau_a = 0.8$). (b) Normalized Recovery loss before/after replay (dashed: prediction-time reference). (c) Historical-credit reversal across horizons (95% Wilson CIs).

Historical expert credit becomes stale. For each matured record, we compare the expert with the highest responsibility computed from prediction-time outputs against the lowest-loss expert under the current expert versions. A historical-credit reversal occurs when the two differ. Across 2,000 records sampled across all datasets, horizons, and seeds, Figure 3(a) shows that the reversal rate decreases from roughly 70% to 20% as capability alignment increases, with an overall rate of 49.9%. Reversals are also substantially more frequent below $\tau_a = 0.8$. The monotone trend supports capability alignment as a useful signal for detecting when prediction-time ownership has become inconsistent with the current expert version.

Targeted replay reduces loss on selected Recovery evidence. For low-alignment items whose current predictive performance has degraded, Figure 3(b) compares expert loss before and after targeted replay, normalized by the prediction-time reference on the same item. Across 120 sampled replay events, the median normalized loss decreases from 1.47 to 1.12, and 97.5% of events improve. Because Recovery is triggered only after low alignment is accompanied by harmful performance change, this result verifies that the targeted replay objective reduces the prediction loss of the selected degraded evidence toward its historical reference.

Credit staleness grows with the forecasting horizon. Figure 3(c) groups the same 2,000 records by forecasting horizon. The reversal rate rises from 21.0% at $H = 1$ to 44.7% at $H = 24$ and 84.1% at $H = 48$. Longer horizons delay complete record-level feedback, allowing more expert updates to occur between the original forecast and the eventual validation of its historical ownership. Historical credit is therefore more likely to become stale as the maturation delay grows, consistent with the stronger long-horizon effects observed in the ablation study.

6 Conclusion

We presented PAVE, a progressive evidence adaptation framework for causal multi-step online time-series forecasting with delayed and progressively revealed feedback. PAVE uses evidence according to both its *maturity* and *current validity*: partial feedback supports fast routing adaptation, whereas fully matured feedback establishes historical expert responsibility that is revalidated against current capability before guiding preservation, rebasing, or recovery. Across four regularly sampled multivariate benchmarks and twelve dataset–horizon settings, PAVE achieves the best or tied-best forecasting accuracy in all evaluated settings. Feedback-matched controls on three representative settings further show that earlier access to partial feedback alone does not eliminate PAVE's advantage, while ablations reveal increasingly important contributions from progressive routing, version-aware validation, and preservation–recovery at longer horizons. Mechanistic diagnostics further show that historical ownership can become inconsistent with current capability and provide empirical support for the proposed validation and recovery mechanisms. Future work may extend this evidence-lifecycle perspective to richer expert architectures and to settings with more irregular or asynchronous supervision.

AI USE STATEMENT

In this work, we used generative AI tools solely for language editing and polishing, including improving grammar, clarity, conciseness, and readability of the manuscript. All AI-assisted edits were reviewed and verified by the authors, and no generative AI tools were used to generate experimental data or results. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work focuses on methodological research for online time-series forecasting using publicly available benchmark datasets. It does not involve human subjects, personally identifiable information, or sensitive data collection. We are not aware of any significant ethical, privacy, security, or fairness concerns arising from the methods or experiments presented in this work.

REPRODUCIBILITY STATEMENT

We provide an anonymous code repository containing the implementation of PAVE, the training and evaluation scripts, and the configurations used in our experiments: `https://anonymous.4open.science/r/PAVE0E57`. The experimental setup, datasets, evaluation protocol, and key hyperparameters are described in Section 5.1 and Appendix B. Additional experimental results and implementation details are provided in the appendix. These materials are intended to support independent reproduction of the results reported in this work.

REFERENCES

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Computer Vision – ECCV 2018*, volume 11207 of *Lecture Notes in Computer Science*, pp. 144–161. Springer, 2018. doi: 10.1007/978-3-030-01219-9_9. URL https://doi.org/10.1007/978-3-030-01219-9_9.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11254–11263, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Aljundi_Task-Free_Continual_Learning_CVPR_2019_paper.html.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: A strong, simple baseline. In *Advances in Neural Information Processing Systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/b704ea2c39778f07c617f6b7ce480e9e-Abstract.html.

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 (eds.), *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pp. 15641–15649. AAAI Press, 2025. doi: 10.1609/AAAI.V39I15.33717. URL https://doi.org/10.1609/aaai.v39i15.33717.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Hkf2_sC5FX.

Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9900–9911. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/70d85f35a1fdc0ab701ff78779306407-Paper.pdf.

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 James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 4869–4877. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/542. URL https://doi.org/10.24963/ijcai.2025/542. Main Track.

Wei Fan, Pengyang Wang, Dongkun Wang, Dongjie Wang, Yuanchun Zhou, and Yanjie Fu. Dish-TS: A general paradigm for alleviating distribution shift in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7522–7529, 2023. doi: 10.1609/aaai.v37i6.25914. URL https://doi.org/10.1609/aaai.v37i6.25914.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/farajtabar20a.html.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://www.jmlr.org/papers/v23/21-0998.html.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4), March 2014. ISSN 0360-0300. doi: 10.1145/2523813. URL https://doi.org/10.1145/2523813.

Johan Hallberg Szabadváry. Adaptive conformal inference for multi-step ahead time-series forecasting online. In Simone Vantini, Matteo Fontana, Aldo Solari, Henrik Boström, and Lars Carlsson (eds.), *Proceedings of the Thirteenth Symposium on Conformal and Probabilistic Prediction with Applications*, volume 230 of *Proceedings of Machine Learning Research*, pp. 250–263. PMLR, 09–11 Sep 2024. URL https://proceedings.mlr.press/v230/hallberg-szabadvary24a.html.

Seungha Hong, Sukang Chae, Suyeon Kim, Sanghwan Jang, and Hwanjo Yu. Dynamic multi-period experts for online time series forecasting. In Hakim Hacid, Yoelle Maarek, Francesco Bonchi, Ido Guy, and Emine Yilmaz (eds.), *Proceedings of the ACM Web Conference 2026, WWW 2026, Dubai, United Arab Emirates, originally scheduled for April 13-17, 2026, rescheduled for June 29 - July 3, 2026*, pp. 7669–7678. ACM, 2026. doi: 10.1145/3774904.3792731. URL https://doi.org/10.1145/3774904.3792731.

Chenfeng Huang, Zixuan Ma, and George Michailidis. Model-agnostic online certificate-driven calibration for time series forecasting under distribution shift. In Emilija Perković and Daniel Malinsky (eds.), *Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence*, volume 337 of *Proceedings of Machine Learning Research*, pp. 2244–2273. PMLR, 17–21 Aug 2026a. URL https://proceedings.mlr.press/v337/huang26b.html.

Xiannan Huang, Shuhan Qiu, Jiayuan Du, and Chao Yang. Online time series prediction using feature adjustment. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 150718–150744, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/f418594e90047a10f4c158f70d6701cc-Paper-Conference.pdf.

HyunGi Kim, Siwon Kim, Jisoo Mok, and Sungroh Yoon. Battling the non-stationarity in time series forecasting via test-time adaptation. In Toby Walsh, Julie Shah, and Zico Kolter (eds.), *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial*

Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025, pp. 17868–17876. AAAI Press, 2025. doi: 10.1609/AAAI.V39I17.33965. URL https://doi.org/10.1609/aaai.v39i17.33965.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *The Tenth International Conference on Learning Representations*. OpenReview.net, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://doi.org/10.1073/pnas.1611835114.

Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997. ISSN 0890-5401. doi: 10.1006/inco.1996.2612. URL https://www.sciencedirect.com/science/article/pii/S0890540196926127.

Ying-yee Ava Lau, Zhiwen Shao, and Dit-Yan Yeung. Fast and slow streams for online time series forecasting without information leakage. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 28736–28762, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/46e624c244cff669223d488defd4e835-Paper-Conference.pdf.

Thomas L. Lee, William Toner, Rajkarn Singh, Artjom Joosen, and Martin Asenov. Lightweight online adaption for time series foundation model forecasts. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 33736–33764. PMLR, 2025. URL https://proceedings.mlr.press/v267/lee25ag.html.

Xingwang Li, Fei Teng, Cong Zhou, and Qiang Duan. Beyond uniform updates: Drift pattern aware online time series forecasting under delayed feedback. In Diego Calvanese (ed.), *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI-26*, pp. 4518–4526. International Joint Conferences on Artificial Intelligence Organization, 8 2026. doi: 10.24963/ijcai.2026/503. URL https://doi.org/10.24963/ijcai.2026/503. Main Track.

Zhizhong Li and Derek Hoiem. Learning without forgetting. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, volume 9908 of *Lecture Notes in Computer Science*, pp. 614–629. Springer, 2016. doi: 10.1007/978-3-319-46493-0_37. URL https://doi.org/10.1007/978-3-319-46493-0_37.

Daojun Liang, Haixia Zhang, Jing Wang, Dongfeng Yuan, and Minggao Zhang. Act now: A novel online forecasting framework for large-scale streaming data. *CoRR*, abs/2412.00108, 2024. doi: 10.48550/ARXIV.2412.00108. URL https://doi.org/10.48550/arXiv.2412.00108.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary Transformers: Exploring the stationarity in time series forecasting. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 9881–9893. Curran Associates, Inc., 2022. doi: 10.52202/068431-0718. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/4054556fcaa934b0bf76da52cf4f92cb-Paper-Conference.pdf.

David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf.

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, João Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2019. doi: 10.1109/TKDE.2018.2876857. URL https://doi.org/10.1109/TKDE.2018.2876857.

L. Melgar-García, A.M. Chacón-Maldonado, G. Asencio-Cortés, and A. Troncoso. A survey on machine learning for streaming time series forecasting. *Computer Science Review*, 62: 101031, 2026. ISSN 1574-0137. doi: 10.1016/j.cosrev.2026.101031. URL https://www.sciencedirect.com/science/article/pii/S1574013726001395.

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*. OpenReview.net, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

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

Dalin Qin, Yehui Li, Weiqi Chen, Zhaoyang Zhu, Qingsong Wen, Liang Sun, Pierre Pinson, and Yi Wang. Evolving multi-scale normalization for time series forecasting under distribution shifts. *CoRR*, abs/2409.19718, 2024. doi: 10.48550/arXiv.2409.19718. URL https://arxiv.org/abs/2409.19718.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Abstract.html.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=3AOj0RCNC2.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*. OpenReview.net, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019. ISSN 2522-5839. doi: 10.1038/s42256-019-0080-x. URL https://doi.org/10.1038/s42256-019-0080-x.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995. PMLR, 2017. URL https://proceedings.mlr.press/v70/zenke17a.html.

Tianxiang Zhan, Ming Jin, Yuanpeng He, Yuxuan Liang, Yong Deng, and Shirui Pan. Continuous evolution pool: Taming recurring concept drift in online time series forecasting. *CoRR*, abs/2506.14790, 2025. doi: 10.48550/ARXIV.2506.14790. URL https://doi.org/10.48550/arXiv.2506.14790.

Yifan 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 A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 69949–69980. Curran Associates, Inc., 2023. doi: 10.52202/075280-3066. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/dd6a47bc0aad6f34aa5e77706d90cdc4-Paper-Conference.pdf.

Yifan Zhang, Weiqi Chen, Zhaoyang Zhu, Dalin Qin, Liang Sun, Xue Wang, Qingsong Wen, Zhang Zhang, Liang Wang, and Rong Jin. Addressing concept shift in online time series forecasting: Detect-then-adapt. *CoRR*, abs/2403.14949, 2024. doi: 10.48550/ARXIV.2403.14949. URL https://doi.org/10.48550/arXiv.2403.14949.

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*, KDD '25, pp. 2020–2031, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400712456. doi: 10.1145/3690624.3709210. URL https://doi.org/10.1145/3690624.3709210.

A Extended Related Work

Online time series forecasting under concept drift. Concept drift broadly concerns changes in streaming data distributions and the resulting need for detection and adaptation (Gama et al., 2014; Lu et al., 2019). Forecasting methods such as RevIN, Non-stationary Transformers, Dish-TS, and EvoMSN address distribution shift or non-stationarity, but do not assume the same online continual-adaptation setting as PAVE (Kim et al., 2022; Liu et al., 2022; Fan et al., 2023; Qin et al., 2024). Online time series forecasting (OTSF) aims to continually adapt a forecasting model as new observations arrive, with concept drift being a central challenge. FSNet introduces complementary fast and slow learning mechanisms to rapidly adapt to recent changes while retaining recurring temporal patterns (Pham et al., 2023). OneNet approaches the problem from an ensemble perspective and dynamically combines forecasters with heterogeneous temporal and cross-variable inductive biases (Zhang et al., 2023). D3A explicitly detects concept shifts and triggers stronger adaptation once a drift is identified (Zhang et al., 2024). More recently, LSTD disentangles long- and short-term latent states, imposing different constraints to preserve persistent dependencies while allowing transient states to change (Cai et al., 2025). LEAF similarly distinguishes long-term macro-drift from short-term, sample-specific micro-drift and addresses them through meta-learned extrapolation and adjustment, respectively (Chen et al., 2025). These methods demonstrate that effective OTSF requires adaptation mechanisms operating at different temporal scales. PAVE is complementary to this line of work: rather than defining adaptation scales solely from the dynamics of the data or model states, it organizes adaptation according to the maturity and validity of the supervision that becomes available after each forecast.

Delayed and progressively available feedback. A distinct difficulty in multi-step OTSF is that future targets are unavailable when a forecast is issued and become observable only after a delay. PROCEED explicitly identifies the temporal gap between the latest supervised model update and the current forecasting distribution and proactively adapts model parameters to compensate for this delay (Zhao & Shen, 2025). DSOF further shows that conventional OTSF protocols can introduce information leakage and reformulates online forecasting so that only genuinely unobserved future targets are evaluated (Lau et al., 2025). Under this protocol, its fast stream learns from incomplete prediction–target pairs, while its slow stream uses fully matured feedback through experience replay. TAFAS studies a related problem from the perspective of test-time adaptation and uses partially observed ground truth to calibrate a pretrained forecasting model under distribution shift (Kim et al., 2025). Act-Now also separates fast and slow update streams: its fast buffer uses partial labels together with consistent pseudo-labels, whereas its slow buffer uses delayed complete labels (Liang et al., 2024). ELF likewise uses online feedback for lightweight adaptation of time-series foundation-model forecasts (Lee et al., 2025).

More recent work has examined how delayed feedback itself should shape online adaptation. PADRE interprets delayed residuals as evidence of latent drift and combines the current context, recent residual trajectories, and retrieved drift patterns to determine how strongly the forecasting model should be updated (Li et al., 2026). ADAPT-Z instead addresses delayed supervision through feature-space adaptation, combining current latent representations with historical gradient information to update forecasting features despite the feedback delay (Huang et al., 2026b). These methods focus primarily on *how* model parameters or representations should be adapted under delayed feedback. PAVE instead uses progressively revealed feedback to determine *what role* the evidence can safely play: partial observations update routing, whereas fully matured records establish historical expert ownership, which is subsequently revalidated as the experts themselves evolve.

Related online forecasting work also studies uncertainty and calibration. Adaptive Conformal Inference extends conformal prediction to multi-step online forecasting and provides finite-sample coverage control across forecast horizons (Hallberg Szabadváry, 2024). Online Certificate-Driven Calibration provides a model-agnostic calibration framework under distribution shift using martingale PAC-Bayesian guarantees (Huang et al., 2026a). These approaches are complementary to PAVE: they primarily target predictive calibration or coverage, whereas PAVE focuses on point-forecast adaptation and the lifecycle of progressively matured evidence across evolving experts.

PAVE likewise treats delayed and partial feedback as useful online evidence, but differs in how that evidence is organized across the adaptation process. Rather than treating partial and complete supervision primarily as two sources for parameter updates, PAVE associates them with different levels of *credit reliability*. Newly matured horizon-level observations provide timely local evidence for rapidly correcting expert routing, while a complete forecasting record provides a more reliable basis for sample-level expert ownership and expert adaptation. This formulation makes feedback maturity an explicit part of the credit-assignment process.

Dynamic experts and recurring concepts. Expert-based approaches offer another way to cope with heterogeneous and recurring regimes. OneNet demonstrates that forecasting models with different inductive biases can be dynamically weighted to improve robustness under concept drift (Zhang et al., 2023). DynaME further distinguishes recurring drift from emergent drift: specialized experts are fitted to relevant historical periodic patterns for recurring regimes, while a stable general expert is emphasized when an uncertain new regime emerges (Hong et al., 2026). Continuous Evolution Pool (CEP) explicitly targets recurring concept drift by maintaining multiple forecaster instances associated with different concepts, retrieving a suitable historical forecaster when a concept reappears and evolving new forecasters when necessary (Zhan et al., 2025).

These approaches primarily preserve or select models associated with different historical regimes. PAVE instead considers a setting in which the experts themselves are continuously updated. This creates an additional problem: credit assigned to expert e when a sample was predicted describes the *prediction-time version* of that expert and may no longer characterize its current capability after subsequent updates. PAVE therefore separates *historical ownership* from *current capability validity*. Historical evidence is used for long-term expert management only after its responsible expert has been re-evaluated against its prediction-time capability. This version-aware treatment allows PAVE to distinguish capability retention, beneficial or neutral evolution, and harmful degradation without maintaining an independent full model for every historical concept.

Continual learning and capability preservation. PAVE is also related to continual-learning methods designed to mitigate catastrophic interference. Replay-based approaches retain representative historical samples and use them to constrain future learning (Rolnick et al., 2019; Buzzega et al., 2020). Gradient Episodic Memory (GEM), for example, projects updates so that the losses of previously observed tasks represented by episodic memories do not increase (Lopez-Paz & Ranzato, 2017), while A-GEM reduces the computational cost by replacing multiple task-specific constraints with a reference gradient computed from episodic memory (Chaudhry et al., 2019). Regularization-based methods instead protect important parameters or preserve prior outputs (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018; Li & Hoiem, 2016). Other approaches protect previous knowledge through gradient geometry rather than direct replay. Orthogonal Gradient Descent (OGD) restricts new gradients to directions that minimally change predictions on previously learned data (Farajtabar et al., 2020). Orthogonal Weight Modification (OWM) similarly restricts parameter changes to reduce interference across contexts (Zeng et al., 2019). Continual Learning in Low-rank Orthogonal Subspaces separates different tasks into low-dimensional orthogonal subspaces to reduce mutual interference (Chaudhry et al., 2020). Gradient Projection Memory (GPM) identifies important activation subspaces from previous tasks and projects subsequent gradients to protect the corresponding learned representations (Saha et al., 2021).

PAVE follows the general intuition that updates along directions supporting useful historical capabilities should be treated differently from unconstrained directions, but departs from conventional task-based continual learning in three aspects. First, there are no explicit task boundaries in the online forecasting stream (Aljundi et al., 2019). Second, historical samples are not assumed to remain equally relevant indefinitely; their ownership and capability validity are continually reassessed. Third, the protected subspace is expert-specific and constructed from responsibility-weighted stable

evidence rather than from all historical activations uniformly. Its protection strength further depends on the accumulated mass of validated evidence. Evidence whose associated capability is genuinely degraded is handled through targeted recovery instead of being permanently protected, while changed capabilities that preserve or improve forecasting performance are rebased rather than restored.

Online learning with expert advice. The fast routing mechanism in PAVE is related to classical online learning with expert advice, where a learner sequentially combines predictions from multiple experts and updates their weights according to observed losses (Cesa-Bianchi & Lugosi, 2006). Multiplicative and exponentiated-gradient updates provide a canonical mechanism for adapting a probability distribution over experts and admit regret guarantees against the best fixed comparator (Kivinen & Warmuth, 1997). OneNet has previously connected this perspective with online time series ensembling by combining online convex optimization with a learned weighting mechanism (Zhang et al., 2023).

PAVE adopts this online-learning perspective only for its routing component. A neural router supplies a context-dependent prior over experts, while a lightweight correction state incorporates progressively revealed expert losses to rapidly revise this prior. This decomposition motivates the no-regret analysis of the idealized progressive routing core against the best fixed convex mixture of the realized prediction-time experts. In contrast, the continually trained neural experts, version-aware memories, and evidence-conditioned preservation and recovery mechanisms remain outside this online-convex abstraction.

Positioning of PAVE. Overall, prior work has separately investigated fast–slow online adaptation, delayed or partial supervision, expert specialization and recurring concepts, and protection against catastrophic interference. PAVE connects these directions through a single *progressive evidence lifecycle*. Feedback maturity determines when evidence is sufficiently informative for local routing credit or sample-level expert ownership; capability validity determines whether that historical ownership still represents the current expert; and the resulting version-aware evidence determines whether a capability should be preserved, rebased, or recovered. This perspective is the main distinction between PAVE and methods that treat partial feedback, expert selection, historical memory, or gradient protection as independent adaptation mechanisms.

B Experimental and Implementation Details

B.1 Datasets and preprocessing

We evaluate PAVE on four multivariate time-series benchmarks: ETTh2, ETTm1, Weather (WTH), and Electricity (ECL). All experiments use the multivariate forecasting setting, an input length of $L = 60$, and prediction horizons $H \in \{1, 24, 48\}$. Table 3 summarizes the dataset configuration used by our implementation.

We preserve chronological order throughout preprocessing, offline training, validation, and online evaluation. ETTh2 and ETTm1 use the fixed boundaries implemented by the ETT data loaders: the first 2880 observations form the training segment, the following 720 observations form the validation segment, and the test stream extends from index 3600 to 14400. Validation and test inputs are allowed to use the preceding $L = 60$ observations as context, but their forecasting targets remain within their respective chronological segments. WTH and ECL use chronological $20\%/5\%/75\%$ train/validation/test splits.

For every dataset, normalization statistics are estimated exclusively from the training segment. Each variable is standardized independently using its training-set mean and standard deviation, and the same transformation is then applied to validation and test observations. No statistics from the online test stream are used to refit the scaler.

B.2 Online evaluation protocol

All methods are evaluated under a rolling chronological stream with forecast origins separated by one timestamp. PAVE follows the leakage-free progressive feedback protocol described in Section 3.

Table 3: **Dataset statistics and forecasting settings.**

Dataset	Variables	Frequency	Input length	Horizons
ETTh2	7	1 hour	60	1/24/48
ETTm1	7	15 min	60	1/24/48
WTH	12	1 hour	60	1/24/48
ECL	321	1 hour	60	1/24/48

Algorithm 1 Online adaptation with PAVE

Require: Experts $\{f_e\}_{e=1}^{E}$, neural router, correction $\mathbf{z}$, Stable memories $\{\mathcal{S}_e\}$, Recovery memories $\{\mathcal{R}_e\}$

for each online origin t **do**
  Observe $\mathbf{y}_t$ and decay $\mathbf{z}$
  Release newly matured partial feedback $\mathcal{F}_t$
  for each $(o, h) \in \mathcal{F}_t$ **do**
    Update $\mathbf{z}$ from the centered mixture-weight gradient
  end for
  Update the neural router from the newly matured local evidence
  for each record o that becomes fully matured **do**
    Compute $\mathbf{r}_o$, κ_o, and capability alignment
    Validate historical ownership using the current Expert versions
    Update Experts under the current Stable subspaces
    Perform targeted Recovery replay and reassess replayed evidence
    Commit validated evidence to Stable or Recovery Memory
  end for
  if periodic reassessment is triggered **then**
    Reassess Stable and Recovery memories
    Rebuild Expert-specific Stable subspaces
  end if
  Combine the neural prior with $\mathbf{z}$ to obtain $\mathbf{p}_t$
  Predict $\hat{\mathbf{Y}}_t$ with the Expert mixture
  Store prediction-time quantities for subsequent feedback
end for

At online origin t, the newly observable target $\mathbf{y}_t$ is first released to all historical records for which it corresponds to a newly matured horizon. Adaptation from this feedback is completed before the forecast at the current origin is issued.

The execution order of PAVE at each origin is:

1. observe the newly available value $\mathbf{y}_t$ and decay the online correction state $\mathbf{z}$ once;
2. release all newly matured horizon-level feedback associated with previous origins;
3. update $\mathbf{z}$ and the neural router from the newly released partial feedback;
4. process records that become fully matured, including sample-level responsibility assignment and version-aware capability validation;
5. update the Experts using fully matured supervision under the previously constructed Stable subspaces, together with targeted Recovery replay when available;
6. commit newly validated evidence to Stable or Recovery Memory and, when scheduled, reassess existing evidence and rebuild the predictive subspaces; and
7. compute the current routing distribution, issue the forecast at origin t, and store its prediction-time outputs and capability sketches for subsequent feedback.

Thus, targets belonging to the forecast currently being generated are never used for its routing or parameter updates. In particular, complete future windows stored by the dataloader are used only for offline evaluation until their individual timestamps become observable through the online stream. Unmatured records are not artificially flushed at the end of the test stream.

Algorithm 1 summarizes this procedure.

B.3 BASELINE IMPLEMENTATIONS

We compare against DSOF, DynaME, ER, FSNet, OneNet, PROCEED, and PatchTST-DGrad. To ensure that differences in data handling do not confound the comparison, all baselines are integrated into the same local data interface and rolling chronological evaluation framework. Our implementations are adaptations of the corresponding public implementations rather than complete mirrors of their original repositories. In particular, FSNet follows the Salesforce FSNet implementation,[1] OneNet and the PatchTST backbone follow the public OneNet implementation,[2] DynaME follows its public implementation,[3] PROCEED follows the OnlineTSF repository,[4] and DSOF follows its public implementation.[5]

All baseline models use the same input length, prediction horizons, dataset splits, offline training budget, and chronological test stream as PAVE. Offline models are trained for at most 15 epochs with batch size 32 and early-stopping patience 3. The offline learning rate is 10^{-3} for ETTh2, ETTm1, and WTH and 3×10^{-3} for ECL.

The main baseline comparison uses delayed causal feedback: an online update can only use a historical forecasting record after the required target has become observable. For the feedback-matched control experiment in Section 5, DSOF, DynaME, and ER use the same causal horizon-level feedback schedule as PAVE. For a forecast issued at origin o, its target at horizon h becomes available only at time $o + h$. At each online origin, all newly matured horizon-level targets from previous forecasts are released before the current forecast is issued. Only these newly observed targets may contribute to the corresponding online update; unrevealed future targets are never used.

For each baseline, the released horizon-level events are passed to its online adaptation procedure using only the observed horizon entries. When multiple feedback events become available at the same origin, their partial losses are aggregated before the online update, so that additional feedback events at the same timestamp do not introduce additional optimizer steps solely because of the number of matured records. The baseline-specific model state and adaptation mechanism are otherwise retained.

This control therefore matches PAVE in the timing and causal availability of partial supervision, while leaving the way each baseline uses that supervision unchanged. It does not introduce PAVE's feedback-driven routing correction, local expert responsibility, version-aware validation, Stable/Recovery memories, or subspace protection into the compared methods.

The main method-specific settings are summarized below. ER uses a PatchTST backbone with a replay buffer of 128 samples and replay batches of size 16. DSOF uses the same slow forecasting backbone together with a width-16 fast student, a student learning rate of 10^{-3}, replay capacity 128, replay batch size 16, and temporal-distillation weight 0.2. DynaME uses two period experts, kernel-ridge regularization 10^{-4}, 8 KRR training samples, a historical window of 672, $\beta = 0.3$, $\delta = 0.01$, and an online learning rate of 10^{-3}. PROCEED uses concept dimension 200, bottleneck dimension 32, and online adapter learning rate 10^{-3}. PatchTST-DGrad uses patch length 16, stride 8, model dimension 32, 8 attention heads, 2 encoder layers, feed-forward dimension 128, RevIN, online learning rate 10^{-3}, and gradient clipping at 1.0. The shared online learning rate for the FSNet, OneNet, and ER implementations is 10^{-3}.

B.4 PAVE ARCHITECTURE AND ROUTING CONFIGURATION

PAVE contains $E = 4$ heterogeneous Experts. With the `mixed` configuration used throughout the main experiments, two Experts follow the FSNet variate-oriented architecture and two follow the FSNet-Time channel-oriented architecture. Both Expert types use a 320-dimensional hidden representation. The FSNet Expert maps the final encoded representation through a joint prediction head, whereas FSNet-Time applies a shared prediction head independently to channel representations.

[1] https://github.com/salesforce/fsnet
[2] https://github.com/yfzhang114/OneNet
[3] https://github.com/shhong97/DynaME
[4] https://github.com/SJTU-DMTai/OnlineTSF
[5] https://github.com/yyalau/iclr2025_dsof

The neural router has a separate shared feature encoder. The input series and seven calendar features are first projected to dimension 320 and processed by a one-layer Transformer encoder with 8 attention heads, feed-forward dimension 640, GELU activation, and dropout 0.05. For the horizon–channel configuration, the router forms its logits from the sum of a channel-specific term, a horizon-specific term, and a learned horizon bias. A softmax temperature of 2.0 is used.

The neural prior remains dense before combination with the feedback-driven correction. At prediction time, effective routing weights are obtained from

$$\mathbf{p}_{t,h,c} = \mathrm{softmax}\left(\log \boldsymbol{\pi}_{t,h,c} + \mathbf{z}_{h,c}\right).$$

The main configuration uses routing top-$K_r = 4$, which is equal to the total number of Experts and therefore leaves every Expert available to the mixture. In contrast, long-term sample-level ownership uses credit top-$K_c = 1$.

The online correction learning rate is 0.1. Its state is decayed once per forecast origin with decay coefficient 0.01. Elementwise correction gradients are clipped at 10, and correction logits are clipped to $[-5, 5]$ before use. Local and sample-level responsibility temperatures are both 1.0. The local-credit term in the partial-feedback router objective has weight 0.1, while the entropy regularization weight is 10^{-3}.

The online neural-router base learning rate is 10^{-5} on all datasets. Router gradients are clipped to norm 0.5.

At prediction time, every Expert representation is mapped through an Expert-specific fixed Gaussian random projection from dimension 320 to dimension 32 and then ℓ_2 normalized. The projection is initialized once using seed 2025 and remains fixed throughout online evaluation. Thus, changes in capability alignment arise from changes in Expert representations rather than from changes in the measurement basis.

B.5 Version-aware memory and targeted recovery

Sample-level responsibility is computed only after a record has fully matured. We use responsibility temperature 1.0, minimum responsibility threshold 0.3, and credit top-$K_c = 1$. Capability validity is determined using the alignment threshold

$$\tau_a = 0.8.$$

Each Expert maintains a Stable Memory and a Recovery Memory, each with capacity 32. Stored tensors are kept in FP16 on CPU. Stable Memory additionally uses a capability-sketch cosine-similarity threshold of 0.98 for duplicate detection. When a duplicate or full-buffer replacement is required, Stable items are ranked by their Stable credit. Recovery items are ranked according to

$$\frac{q_{i,e}^{\mathrm{rec}}}{1 + 0.5\, n_{i,e}^{\mathrm{att}}},$$

where $n_{i,e}^{\mathrm{att}}$ is the number of previous Recovery attempts.

For direction-aware validation, harmful drift is declared when the current prediction loss exceeds its current reference according to

$$\ell_{i,e}^{\mathrm{cur}} > (1+\delta)\ell_{i,e}^{\mathrm{ref}} + \epsilon,$$

with degradation margin $\delta = 0$ and numerical tolerance $\epsilon = 10^{-8}$. Low-alignment changes that do not satisfy this degradation criterion are treated as beneficial or neutral evolution, and their capability reference is rebased to the current sketch and current loss rather than sent to Recovery.

During each fully matured Expert update, at most two Recovery items are selected per Expert. Recovery sampling is deterministic and prioritizes the score above, while excluding duplicate sample IDs within the same update. The replay objective is weighted by 0.1, and the sketch-distillation term inside each Recovery objective has coefficient 1.0. Prediction-time reference sketches are detached from the computation graph.

Recovery items are reassessed immediately after replay. An item whose harmful performance degradation has disappeared is rebased and returned to Stable Memory when accepted by the Stable buffer.

The implementation additionally uses an alignment promotion threshold of 0.9 and loss threshold 1.0 as a fallback recovery criterion. A Recovery item is discarded after three unsuccessful replay attempts. Stable and Recovery memories are also globally reassessed every 100 fully matured records.

B.6 Responsibility-weighted subspace protection

Stable evidence defines an Expert-specific predictive subspace over the exact features consumed by its final prediction head. The Stable covariance geometry is constructed from the uncentered weighted second moment

$$\mathbf{M}_e = \frac{\sum_i q_{i,e}^{\mathrm{sta}} \mathbf{h}_{i,e} \mathbf{h}_{i,e}^\top}{\sum_i q_{i,e}^{\mathrm{sta}}}.$$

For FSNet-Time Experts, channel-level head features are treated as separate observations while the sample's Stable credit is divided across its channels, so their total weight remains equal to that sample's Stable credit.

The main configuration fixes the predictive-subspace rank to 16. At least four Stable samples are required before refreshing the basis, and the basis is recomputed every 100 fully matured records when sufficient evidence exists. The implementation supports at most rank 32; the 0.95 cumulative-energy criterion is used only when automatic rank selection is enabled rather than the fixed-rank setting used in our experiments.

Subspace protection is restricted to prediction-head weight gradients: `ExpertNet.regressor` for FSNet Experts and `FSNetTimeExpertNet.regressor_time` for FSNet-Time Experts. Encoder parameters and prediction-head biases are updated without projection.

Let

$$m_e = \sum_i q_{i,e}^{\mathrm{sta}}, \qquad \mu_e = \frac{m_e}{m_e + m_0},$$

where the evidence-mass scale is $m_0 = 8$. The implementation uses $\lambda_{\mathrm{sub}} = 10^4$ and computes

$$\gamma_e = \frac{1}{1 + \eta_e \lambda_{\mathrm{sub}} \mu_e},$$

clipped to $[0, 1]$, where η_e is the current Expert learning rate. When Stable Memory becomes empty, the cached basis may be retained for efficiency but its evidence mass is reset to zero, yielding $\gamma_e = 1$ and therefore no effective gradient suppression.

B.7 Optimization and hyperparameters

PAVE is pretrained offline before chronological online evaluation. Expert and router parameters use separate AdamW optimizers with weight decay 10^{-3}. The maximum offline training budget is 15 epochs with batch size 32 and early-stopping patience 3. The learning-rate schedule halves the offline learning rate after each epoch. The training loss uses MSE; Expert and router gradient norms are clipped at 1.0 and 0.5, respectively.

The offline Expert and router learning rates are both 10^{-3} for ETTh2, ETTm1, and WTH, and 3×10^{-3} for ECL. During online adaptation, the Expert base learning rate is dataset dependent:

$$10^{-4} \text{ for ETTh2 and ETTm1}, \qquad 5 \times 10^{-5} \text{ for WTH}, \qquad 10^{-5} \text{ for ECL}.$$

The online router base learning rate is 10^{-5} for all datasets. The online test batch size is one, and one inner optimization step is performed for each fully matured record.

The implementation uses an error-adaptive controller for online Expert and router learning rates. Let ℓ_t be the most recent online mixture MSE and $\bar{\ell}_t$ its exponential moving average with coefficient 0.9. The learning-rate scale is

$$s_t = \mathrm{clip}\left(\frac{1}{\sqrt{\ell_t / (\bar{\ell}_t + \epsilon)}}, 0.1, 1 \right),$$

and both online learning rates are multiplied by s_t. Hence the adaptive controller can make updates more conservative after an error spike but never increase either learning rate above its configured dataset-specific base value.

Table 4 summarizes the main PAVE configuration.

Table 4: **Main PAVE hyperparameters used in the experiments.**

Hyperparameter	Value
Number of Experts E	4
Expert composition	2 FSNet + 2 FSNet-Time
Router type	Horizon–channel
Router temperature	2.0
Routing top-K_r	4
Credit top-K_c	1
Router entropy weight	10^{-3}
Local-credit weight	0.1
Correction learning rate	0.1
Correction decay	0.01
Correction gradient clip	10
Correction logit clip	5
Responsibility temperature	1.0
Responsibility threshold	0.3
Alignment threshold τ_a	0.8
Capability sketch dimension	32
Capability sketch seed	2025
Stable Memory capacity	32 / Expert
Recovery Memory capacity	32 / Expert
Stable duplicate threshold	0.98
Recovery batch size	2 / Expert
Recovery loss weight	0.1
Sketch-distillation weight	1.0
Recovery failure penalty	0.5
Maximum Recovery attempts	3
Memory reassessment interval	100 records
Subspace rank	16
Subspace minimum samples	4
Subspace refresh interval	100 records
Subspace λ_{sub}	10^4
Evidence-mass scale m_0	8
Expert gradient clip	1.0
Router gradient clip	0.5
Weight decay	10^{-3}
Offline batch size	32
Online batch size	1
Offline epochs	15 (maximum)
Early-stopping patience	3

B.8 Ablation Configurations

All ablations use the same architecture, progressive-feedback protocol, offline initialization, and online optimization settings as full PAVE, except for the component explicitly modified below. Table 5 summarizes their operational definitions.

These definitions are chosen to isolate distinct roles in the evidence lifecycle. In particular, *w/o Version Awareness* removes the expert-version check altogether, whereas *w/o Direction Awareness* retains the alignment check but removes the loss-direction decision used to separate beneficial or neutral evolution from harmful degradation. Similarly, *Unweighted Subspace* does not remove responsibility from memory admission or protection strength; it removes responsibility weighting only from the estimation of the protected subspace.

B.9 Computational environment

All experiments are implemented in PyTorch and executed on a heterogeneous GPU cluster consisting of two NVIDIA A100-SXM4-80GB GPUs and three NVIDIA RTX 4090 GPUs. The software environment uses Python 3.10.21, PyTorch 2.5.1 with CUDA 12.4 support, and NVIDIA driver 550.127.05. Each individual run uses a single GPU, while independent dataset–horizon configu-

Table 5: Operational definitions of the ablations in Figure 2. All components not listed as modified are kept identical to full PAVE.

Variant	Operational modification
w/o Online Correction	Disable the feedback-driven correction state $\mathbf{z}$. Newly observed targets no longer update $\mathbf{z}$, and routing uses the neural-router prior alone. Partial-feedback router training, including the partial mixture loss and local-credit supervision, remains enabled.
w/o Local Credit	Set $\lambda_{\mathrm{loc}} = 0$ in equation 8, removing the confidence-weighted cross-entropy supervision from local responsibility. The partial mixture loss, router entropy regularization, and feedback-driven correction $\mathbf{z}$ remain unchanged.
w/o Version Awareness	Disable capability-alignment validation by setting the effective alignment of eligible matured evidence to one. Historical ownership is therefore admitted as Stable evidence without version-based reassessment. Direction-aware classification and capability rebasing are consequently inactive, while the remaining routing and Stable-subspace mechanisms are unchanged.
w/o Direction Awareness	Retain capability-alignment validation but remove the predictive loss-direction test after low alignment is detected. Consequently, high-alignment evidence remains Stable, whereas low-alignment evidence is eligible for Recovery irrespective of whether its current predictive loss has improved or degraded. Beneficial/neutral capability rebasing is disabled.
w/o Subspace Protection	Remove Stable-subspace gradient filtering and update Experts with their raw online gradients. The Stable/Recovery memories, version-aware lifecycle, and Recovery replay remain active.
Unweighted Subspace	Retain Stable-subspace protection but replace the Stable-evidence weights in the second-moment matrix with uniform sample weights. Memory admission, version-aware reassessment, Recovery, and the evidence-adaptive protection strength γ_e remain unchanged; only the geometry of the protected subspace is made unweighted.
w/o Recovery	Disable the Recovery Memory and targeted replay branch. Direction-aware validation remains active: beneficial or neutral low-alignment evolution can still be rebased, whereas newly detected harmful low-alignment evidence is not admitted for Recovery. Stable-subspace preservation remains enabled.

rations and random seeds may be executed concurrently across multiple GPUs. Multi-seed results are reported over three independent runs using seeds 0, 1, and 2 with the same hyperparameter configuration.

C Regret Analysis of Progressive Routing

This appendix provides the full derivation and proof of Theorem 1, together with its asymptotic corollaries. We analyze the same idealized dense routing core with fixed expert predictions, no correction decay, no clipping, and no top-k truncation. The analysis isolates the interaction between the neural routing prior, the feedback-driven online correction, and delayed supervision. It is therefore a routing-level analysis rather than a regret guarantee for the complete PAVE system, which additionally contains online neural-router updates, evolving experts, version-aware memory, subspace protection, and targeted recovery.

C.1 Setup

We first consider a single routing coordinate, suppressing the horizon and channel indices for clarity. Let E denote the number of experts and

$$\Delta_E = \left\{ \mathbf{p} \in \mathbb{R}_+^E : \sum_{e=1}^{E} p_e = 1 \right\}$$

the probability simplex. At routing round t, expert e produces a scalar prediction $\hat{y}_{t,e}$, and the router forms the mixture prediction

$$\hat{y}_t(\mathbf{p}) = \sum_{e=1}^{E} p_e \hat{y}_{t,e}, \qquad \mathbf{p} \in \Delta_E. \tag{15}$$

Once the corresponding target y_t becomes available, the routing loss is

$$f_t(\mathbf{p}) = \left(\hat{y}_t(\mathbf{p}) - y_t\right)^2. \tag{16}$$

Since $\hat{y}_t(\mathbf{p})$ is affine in $\mathbf{p}$, f_t is convex on Δ_E. We compare the online routing decisions against an arbitrary fixed mixture $\mathbf{u} \in \Delta_E$:

$$R_T(\mathbf{u}) = \sum_{t=1}^{T} \left[f_t(\mathbf{p}_t) - f_t(\mathbf{u})\right]. \tag{17}$$

In particular, $\mathbf{u}$ may be chosen as the best fixed expert mixture in hindsight.

The gradient of equation 16 with respect to the mixture weight of expert e is

$$g_{t,e} = 2\left(\hat{y}_t - y_t\right)\hat{y}_{t,e}. \tag{18}$$

Throughout the analysis, we assume

$$\|\mathbf{g}_t\|_\infty \le G \tag{19}$$

for all t, which follows, for example, when targets and expert predictions are bounded.

Multi-step forecasting produces delayed feedback. Let $d_t \ge 0$ denote the number of routing rounds for which the gradient generated at round t remains unavailable. Equivalently, if q_t denotes the number of outstanding gradients when round t is predicted, then

$$D_T := \sum_{t=1}^{T} q_t = \sum_{t=1}^{T} d_t \tag{20}$$

is the cumulative feedback delay. For a forecasting horizon bounded by H, we have $D_T \le HT$.

C.2 The PAVE Correction Is a Centered Routing Gradient

The online correction used by PAVE is updated from a newly observed target using

$$\tilde{g}_{t,e} = 2\left(\hat{y}_t - y_t\right)\left(\hat{y}_{t,e} - \hat{y}_t\right). \tag{21}$$

At first sight, equation 21 differs from the ordinary mixture-loss gradient in equation 18. However,

$$\tilde{\mathbf{g}}_t = \mathbf{g}_t - 2\left(\hat{y}_t - y_t\right)\hat{y}_t\mathbf{1}, \tag{22}$$

where $\mathbf{1}$ is the all-ones vector. Hence the two gradients differ only by a common additive constant across experts.

For any $\mathbf{p}, \mathbf{u} \in \Delta_E$,

$$\begin{aligned}\langle \tilde{\mathbf{g}}_t, \mathbf{p} - \mathbf{u}\rangle &= \langle \mathbf{g}_t, \mathbf{p} - \mathbf{u}\rangle - c_t\langle \mathbf{1}, \mathbf{p} - \mathbf{u}\rangle \\ &= \langle \mathbf{g}_t, \mathbf{p} - \mathbf{u}\rangle,\end{aligned} \tag{23}$$

because both $\mathbf{p}$ and $\mathbf{u}$ sum to one. The same common shift also leaves a softmax distribution unchanged. Therefore the centered correction implemented by PAVE is equivalent to the ordinary routing gradient for simplex-constrained online optimization.

C.3 Optimistic Entropic Routing Interpretation

We next show how the dense PAVE routing rule can be represented as an optimistic entropic mirror-descent decision.

Let $\mathbf{z}_t \in \mathbb{R}^E$ denote the accumulated feedback-driven correction before round t, and define

$$\boldsymbol{\mu}_t = \mathrm{softmax}(\mathbf{z}_t). \tag{24}$$

In the idealized setting without decay or clipping, every available gradient updates

$$\mathbf{z} \leftarrow \mathbf{z} - \eta\tilde{\mathbf{g}}, \tag{25}$$

where $\eta > 0$ is the correction learning rate.

Let $\boldsymbol{\pi}_t \in \Delta_E$ be the dense neural routing prior, with $\pi_{t,e} > 0$. Because a common additive shift in an entropic linear term does not affect the resulting simplex decision, we use the centered log-prior representation

$$m_{t,e}(\eta) = -\frac{1}{\eta}\left(\log \pi_{t,e} - \frac{1}{E}\sum_{j=1}^{E} \log \pi_{t,j}\right). \tag{26}$$

The centering fixes a canonical representative of the log-prior while leaving the routing distribution unchanged. Using the negative-entropy regularizer, the corresponding optimistic routing decision is

$$\mathbf{p}_t = \arg\min_{\mathbf{p}\in\Delta_E} \left\{\eta \left\langle \mathbf{m}_t(\eta), \mathbf{p}\right\rangle + \mathrm{KL}\left(\mathbf{p} \,\|\, \boldsymbol{\mu}_t\right)\right\}. \tag{27}$$

Solving equation 27 gives

$$\begin{aligned} p_{t,e} &\propto \mu_{t,e} \exp(-\eta m_{t,e}(\eta)) \\ &\propto \exp(z_{t,e})\pi_{t,e}, \end{aligned} \tag{28}$$

and therefore

$$\boxed{\mathbf{p}_t = \mathrm{softmax}\left(\log \boldsymbol{\pi}_t + \mathbf{z}_t\right).} \tag{29}$$

This is exactly the dense routing rule used by PAVE before optional top-k selection.

Equation 29 gives a useful interpretation of the two routing components. The neural router supplies a context-dependent prior preference $\boldsymbol{\pi}_t$, while $\mathbf{z}_t$ accumulates corrections obtained from progressively revealed feedback.

Importantly, the induced optimistic vector $\mathbf{m}_t(\eta)$ depends on the stepsize η through equation 26. Consequently, for a general non-uniform neural prior, its mismatch with the realized routing gradient cannot automatically be treated as a stepsize-independent constant. We therefore retain this dependence explicitly in the regret analysis below.

C.4 Delayed Routing Regret

For a fixed stepsize $\eta > 0$, define

$$V_T(\eta) = \sum_{t=1}^{T} \left\|\mathbf{g}_t - \mathbf{m}_t(\eta)\right\|_\infty^2, \tag{30}$$

which measures the mismatch between the prior-induced optimistic signal and the realized routing gradient.

With these definitions, Theorem 1 in the main text states that, for any fixed comparator $\mathbf{u} \in \Delta_E$,

$$R_T(\mathbf{u}) \le \frac{\log E}{\eta} + \frac{\eta}{2} V_T(\eta) + 2\eta G^2 D_T. \tag{31}$$

The first term is the usual entropic initialization cost, the second measures disagreement between the contextual neural prior and the realized routing gradients, and the third is the additional cost induced by delayed feedback. Thus the bound separates the two quantities most relevant to progressive routing: prior quality and feedback delay.

The following corollary gives a sufficient condition under which the finite-horizon bound yields vanishing average regret for a non-uniform neural prior.

Corollary 2 (Vanishing regret under prior–gradient compatibility)**.** *Consider a sequence of horizons T and associated stepsizes η_T. Suppose that there exists a finite constant B, independent of T, such that*

$$V_T(\eta_T) \le B^2 T. \tag{32}$$

If the maximum feedback delay is bounded by H, then

$$R_T(\mathbf{u}) \le \frac{\log E}{\eta_T} + \eta_T T \left(\frac{B^2}{2} + 2G^2 H \right). \tag{33}$$

Hence, under equation 32, taking

$$\eta_T = \Theta \left(\sqrt{\frac{\log E}{T(B^2 + G^2 H)}} \right) \tag{34}$$

yields

$$R_T(\mathbf{u}) = \mathcal{O} \left(\sqrt{T(B^2 + G^2 H) \log E} \right). \tag{35}$$

For fixed B, G, and H,

$$\frac{R_T(\mathbf{u})}{T} \longrightarrow 0.$$

Condition equation 32 is an additional compatibility assumption on the learned prior; it is not implied merely by $\pi_{t,e} > 0$. In particular, because $\mathbf{m}_t(\eta)$ scales with $1/\eta$, a fixed non-uniform prior need not satisfy equation 32 as η_T decreases. The corollary therefore characterizes a sufficient regime in which the contextual prior remains compatible with the realized gradients, rather than providing an unconditional guarantee for an arbitrary learned router.

A useful special case is obtained when the neural prior is uniform.

Corollary 3 (Correction-only routing)**.** *If*

$$\boldsymbol{\pi}_t = \left(\frac{1}{E}, \dots, \frac{1}{E} \right)$$

for all t, then the centered signal in equation 26 is $\mathbf{m}_t(\eta) = \mathbf{0}$, and therefore

$$V_T(\eta) = \sum_{t=1}^{T} \|\mathbf{g}_t\|_\infty^2 \le G^2 T.$$

If the maximum delay is bounded by H, choosing

$$\eta = \Theta \left(\sqrt{\frac{\log E}{TG^2(H+1)}} \right)$$

gives

$$R_T(\mathbf{u}) = \mathcal{O} \left(G\sqrt{T(H+1)\log E} \right),$$

and hence vanishing average regret for fixed H. Thus the feedback-driven correction alone recovers the standard delayed exponentiated-gradient behavior, while the learned prior contributes an additional mismatch term in the general PAVE routing rule.

C.5 Proof of Theorem 1

We separate the argument into the immediate-feedback optimistic mirror-descent term and the additional penalty induced by delayed feedback.

Step 1: Immediate-feedback optimistic mirror descent. Let

$$\bar{\boldsymbol{\mu}}_{t+1} = \arg\min_{\mathbf{p} \in \Delta_E} \{ \eta \langle \mathbf{g}_t, \mathbf{p} \rangle + \mathrm{KL}(\mathbf{p} \,\|\, \boldsymbol{\mu}_t) \} \tag{36}$$

denote the correction-state update that would be performed if the gradient generated at round t were immediately available.

The negative-entropy regularizer is 1-strongly convex with respect to the ℓ_1 norm. The standard optimistic mirror-descent inequality therefore gives, for any $\mathbf{u} \in \Delta_E$,

$$\begin{aligned} \langle \mathbf{g}_t, \mathbf{p}_t - \mathbf{u} \rangle \le{}& \frac{\mathrm{KL}(\mathbf{u} \,\|\, \boldsymbol{\mu}_t) - \mathrm{KL}(\mathbf{u} \,\|\, \bar{\boldsymbol{\mu}}_{t+1})}{\eta} \\ &+ \frac{\eta}{2} \|\mathbf{g}_t - \mathbf{m}_t(\eta)\|_\infty^2. \end{aligned} \tag{37}$$

By convexity of f_t,

$$f_t(\mathbf{p}_t) - f_t(\mathbf{u}) \le \langle \mathbf{g}_t, \mathbf{p}_t - \mathbf{u} \rangle . \tag{38}$$

For the auxiliary immediate-feedback process, $\boldsymbol{\mu}_{t+1} = \bar{\boldsymbol{\mu}}_{t+1}$, so summing equation 37 over t telescopes the KL terms. Initializing the correction state at zero gives a uniform $\boldsymbol{\mu}_1$, and hence

$$\mathrm{KL}\left(\mathbf{u} \,\|\, \boldsymbol{\mu}_1\right) \le \log E.$$

Therefore the corresponding immediate-feedback decisions satisfy

$$R_T^{\mathrm{imm}}(\mathbf{u}) \le \frac{\log E}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\mathbf{g}_t - \mathbf{m}_t(\eta)\|_\infty^2 . \tag{39}$$

Step 2: Stability under missing gradients. Under delayed feedback, the correction state used at round t has not yet incorporated q_t outstanding gradients. Let $\mathbf{p}_t^{\mathrm{del}}$ denote the actual delayed decision and $\mathbf{p}_t^{\mathrm{imm}}$ the auxiliary decision obtained after applying the same realized gradients without delay. Both use the same contextual prior $\boldsymbol{\pi}_t$ at round t, so their difference is caused only by the missing correction updates.

Because a common shift in the gradient does not change the softmax update, we may analyze the equivalent ordinary-gradient update. Applying one gradient $\mathbf{g}$ with $\|\mathbf{g}\|_\infty \le G$ changes the correction logits by at most ηG in ℓ_∞. With q_t missing updates, the difference between the delayed and immediate correction logits is therefore at most $\eta G q_t$ in ℓ_∞.

The softmax map satisfies

$$\|\mathrm{softmax}(\mathbf{v} + \boldsymbol{\delta}) - \mathrm{softmax}(\mathbf{v})\|_1 \le 2\|\boldsymbol{\delta}\|_\infty.$$

Since the same $\log \boldsymbol{\pi}_t$ is added to both sets of correction logits, it follows that

$$\left\|\mathbf{p}_t^{\mathrm{del}} - \mathbf{p}_t^{\mathrm{imm}}\right\|_1 \le 2\eta G q_t. \tag{40}$$

Using Hölder's inequality,

$$\begin{aligned} \left\langle \mathbf{g}_t, \mathbf{p}_t^{\mathrm{del}} - \mathbf{p}_t^{\mathrm{imm}} \right\rangle &\le \|\mathbf{g}_t\|_\infty \left\|\mathbf{p}_t^{\mathrm{del}} - \mathbf{p}_t^{\mathrm{imm}}\right\|_1 \\ &\le 2\eta G^2 q_t. \end{aligned} \tag{41}$$

Summing over all rounds and using $\sum_t q_t = D_T$ yields

$$\sum_{t=1}^{T} \left\langle \mathbf{g}_t, \mathbf{p}_t^{\mathrm{del}} - \mathbf{p}_t^{\mathrm{imm}} \right\rangle \le 2\eta G^2 D_T. \tag{42}$$

Finally, by convexity,

$$\begin{aligned} f_t(\mathbf{p}_t^{\mathrm{del}}) - f_t(\mathbf{u}) &\le \left\langle \mathbf{g}_t, \mathbf{p}_t^{\mathrm{del}} - \mathbf{u} \right\rangle \\ &= \left\langle \mathbf{g}_t, \mathbf{p}_t^{\mathrm{imm}} - \mathbf{u} \right\rangle + \left\langle \mathbf{g}_t, \mathbf{p}_t^{\mathrm{del}} - \mathbf{p}_t^{\mathrm{imm}} \right\rangle . \end{aligned} \tag{43}$$

Combining equation 39 and equation 42 therefore gives

$$R_T(\mathbf{u}) \le \frac{\log E}{\eta} + \frac{\eta}{2} V_T(\eta) + 2\eta G^2 D_T, \tag{44}$$

which establishes equation 31. □

Proof of Corollary 2. Under the compatibility condition $V_T(\eta_T) \le B^2 T$ and the bounded-delay condition $D_T \le HT$, Theorem 1 gives

$$R_T(\mathbf{u}) \le \frac{\log E}{\eta_T} + \eta_T T \left(\frac{B^2}{2} + 2G^2 H \right).$$

Balancing the two terms yields the stepsize in equation 34 up to constant factors, and hence equation 35. Dividing by T proves vanishing average regret for fixed B, G, and H. □

Proof of Corollary 3. For a uniform prior,

$$\log \pi_{t,e} = \frac{1}{E} \sum_{j=1}^{E} \log \pi_{t,j},$$

so equation 26 gives $\mathbf{m}_t(\eta) = \mathbf{0}$. Therefore

$$V_T(\eta) = \sum_{t=1}^{T} \|\mathbf{g}_t\|_\infty^2 \leq G^2 T.$$

The result then follows directly from Corollary 2 with $B = G$. □

C.6 Extension to Horizon–Channel Routing

PAVE maintains a separate correction for every horizon–channel coordinate. For a fixed (h, c) pair, the analysis above applies directly to $\mathbf{p}_{t,h,c} \in \Delta_E$. If the routing loss is summed or averaged over coordinates, the corresponding regret bounds can likewise be summed or averaged over (h, c).

Importantly, the feedback delay depends on the horizon: the target associated with horizon h becomes available after h online steps. Thus the progressive multi-step setting naturally induces the delayed-gradient structure modeled above.

The delay term also clarifies the role of progressively revealed feedback. Waiting until an entire forecasting record has matured would postpone all routing corrections associated with that record to the largest horizon delay. PAVE instead incorporates each horizon-specific correction as soon as the corresponding target becomes available. In the idealized analysis, this reduces the number of outstanding gradients q_t and therefore the cumulative delay D_T relative to a strategy that waits for complete record maturation.

C.7 Relation to the Practical PAVE Implementation

The analysis above intentionally isolates the routing mechanism. We summarize its correspondence to the practical implementation and the resulting scope of the theoretical result.

Neural routing prior. The practical router outputs a dense horizon–channel-dependent prior $\boldsymbol{\pi}_{t,h,c}$. Equation 26 represents this distribution as an optimistic linear term in the entropic routing update. This is an algebraic interpretation of the contextual prior; we do not assume that the neural router explicitly predicts the future loss gradient. For a general learned prior, the effect of its disagreement with the realized gradient is retained explicitly through $V_T(\eta)$.

Online correction. The practical correction is updated from prediction-time expert outputs, the stored mixture prediction, and newly observed targets. Its update direction is exactly the centered mixture-loss gradient in equation 21. Because centering adds only an expert-independent constant, it is equivalent to the ordinary routing gradient on the simplex.

Progressive feedback. The implementation applies a correction whenever an individual horizon target becomes available rather than waiting for the entire forecasting record. This corresponds directly to the delayed-feedback process modeled above and motivates the cumulative-delay term D_T.

Practical deviations. Several implementation choices are excluded from Theorem 1: correction decay, gradient and logit clipping, optional top-k routing, SGD updates of the neural router, and the fact that expert parameters continue to evolve. The Stable/Recovery memories, version-aware validation, subspace protection, and targeted replay are also outside the routing-only analysis. In addition, the practical implementation uses a fixed correction learning rate rather than the horizon-dependent schedule appearing in the asymptotic corollaries.

Most importantly, the learned-prior vanishing-regret statement in Corollary 2 requires the explicit prior–gradient compatibility condition in equation 32. We neither assume nor claim that the learned

Table 6: MAE (mean ± standard deviation over three runs; lower is better). Bold and underline denote the best and second-best mean results, respectively.

	ETTh2			ETTm1		
Method	1	24	48	1	24	48
DSOF	0.431 ± 0.016	0.767 ± 0.069	0.950 ± 0.044	0.232 ± 0.001	0.575 ± 0.003	0.674 ± 0.005
DynaME	$\underline{0.348 \pm 0.034}$	$\underline{0.683 \pm 0.042}$	0.887 ± 0.029	$\underline{0.189 \pm 0.003}$	0.600 ± 0.004	0.673 ± 0.005
ER	0.391 ± 0.007	$\underline{0.683 \pm 0.046}$	$\underline{0.883 \pm 0.037}$	0.202 ± 0.004	0.574 ± 0.009	$\underline{0.671 \pm 0.006}$
FSNet	0.453 ± 0.009	0.797 ± 0.038	0.968 ± 0.019	0.274 ± 0.013	0.783 ± 0.022	0.894 ± 0.030
OneNet	0.443 ± 0.008	0.790 ± 0.067	1.035 ± 0.077	0.246 ± 0.006	0.730 ± 0.002	0.838 ± 0.008
PROCEED	0.383 ± 0.006	0.753 ± 0.046	0.970 ± 0.029	0.195 ± 0.003	$\underline{0.567 \pm 0.007}$	0.673 ± 0.006
PatchTST-DGrad	0.392 ± 0.004	0.743 ± 0.052	0.906 ± 0.034	0.239 ± 0.005	0.617 ± 0.007	0.708 ± 0.005
PAVE	$\mathbf{0.338 \pm 0.003}$	$\mathbf{0.681 \pm 0.002}$	$\mathbf{0.879 \pm 0.009}$	$\mathbf{0.186 \pm 0.002}$	$\mathbf{0.562 \pm 0.013}$	$\mathbf{0.669 \pm 0.002}$

	WTH			ECL		
Method	1	24	48	1	24	48
DSOF	0.237 ± 0.001	0.485 ± 0.002	0.586 ± 0.003	0.258 ± 0.004	0.404 ± 0.005	$\underline{0.453 \pm 0.004}$
DynaME	0.205 ± 0.001	$\underline{0.432 \pm 0.002}$	$\underline{0.538 \pm 0.003}$	0.267 ± 0.004	0.506 ± 0.007	0.634 ± 0.005
ER	0.211 ± 0.002	0.445 ± 0.001	0.544 ± 0.004	$\underline{0.249 \pm 0.005}$	$\underline{0.403 \pm 0.006}$	0.457 ± 0.003
FSNet	0.303 ± 0.003	0.538 ± 0.004	0.634 ± 0.005	1.019 ± 0.022	1.643 ± 0.026	1.751 ± 0.013
OneNet	0.233 ± 0.001	0.523 ± 0.131	0.629 ± 0.155	0.432 ± 0.019	0.612 ± 0.007	0.625 ± 0.027
PROCEED	$\underline{0.197 \pm 0.003}$	0.464 ± 0.001	0.568 ± 0.003	0.286 ± 0.006	0.481 ± 0.004	0.543 ± 0.004
PatchTST-DGrad	0.209 ± 0.001	0.478 ± 0.002	0.573 ± 0.005	0.281 ± 0.006	0.417 ± 0.005	0.477 ± 0.004
PAVE	$\mathbf{0.191 \pm 0.001}$	$\mathbf{0.421 \pm 0.001}$	$\mathbf{0.533 \pm 0.003}$	$\mathbf{0.240 \pm 0.003}$	$\mathbf{0.402 \pm 0.003}$	$\mathbf{0.451 \pm 0.004}$

neural router satisfies this condition throughout the practical online stream. Accordingly, Theorem 1 should be interpreted as a finite-horizon regret decomposition for the *idealized progressive routing core*. Corollary 2 characterizes a sufficient regime for vanishing average regret, while Corollary 3 recovers an unconditional delayed exponentiated-gradient guarantee for the correction-only special case. These results provide supporting online-learning intuition rather than a theoretical guarantee for the complete PAVE algorithm.

D Full Experimental Results

This appendix complements the primary MSE comparison in Table 1 with the corresponding MAE results and reports the complete per-setting results underlying the ablation summary in Figure 2. All results are reported as mean ± standard deviation over three runs.

D.1 Complete Forecasting Metrics

Table 6 reports MAE on the same four datasets and forecasting horizons used in the main MSE comparison. MSE remains the primary metric in the main paper, while MAE provides a complementary measure that is less sensitive to large individual errors. PAVE obtains the lowest mean MAE in all 12 dataset–horizon settings. Together with the MSE results in Table 1, this indicates that the observed improvements are consistent across both squared- and absolute-error criteria.

D.2 Per-Setting Ablation Results

Figure 2 summarizes each ablation by averaging its relative MSE increase over datasets at the same forecasting horizon. Table 7 reports the corresponding raw MSE for every dataset–horizon setting, allowing the aggregate trends to be examined at the dataset level.

The per-setting results show that the degradation observed in Figure 2 is not driven by a single dataset. At the reported precision, every ablation is either worse than or tied with full PAVE in all 12 settings, while every ablation produces higher MSE than PAVE at $H = 48$ on all four datasets. The differences are generally small at $H = 1$ and become substantially larger as the forecasting horizon increases.

This pattern is particularly clear for the components related to historical credit and long-term capability management. Removing version awareness causes large degradation at long horizons across all four datasets, while removing direction awareness, Stable-subspace protection, or Recovery produces a similar but somewhat smaller deterioration. The two partial-feedback routing components

Table 7: Complete ablation results in MSE (mean ± standard deviation over three runs; lower is better). Full PAVE is included as the reference corresponding to Figure 2.

	ETTh2			ETTm1		
Variant	1	24	48	1	24	48
PAVE	$\mathbf{0.355 \pm 0.005}$	$\mathbf{2.408 \pm 0.008}$	$\mathbf{4.919 \pm 0.074}$	$\mathbf{0.085 \pm 0.001}$	$\mathbf{0.557 \pm 0.016}$	$\mathbf{0.759 \pm 0.004}$
w/o Online Correction	0.368 ± 0.004	2.589 ± 0.002	6.071 ± 0.144	0.086 ± 0.001	0.581 ± 0.007	0.795 ± 0.008
w/o Local Credit	0.357 ± 0.006	2.508 ± 0.003	5.634 ± 0.059	0.086 ± 0.002	0.566 ± 0.014	0.797 ± 0.003
w/o Version Awareness	0.358 ± 0.004	2.652 ± 0.016	6.896 ± 0.538	0.085 ± 0.001	0.566 ± 0.016	0.800 ± 0.003
w/o Direction Awareness	0.357 ± 0.004	2.460 ± 0.006	5.787 ± 0.043	0.085 ± 0.000	0.576 ± 0.006	0.804 ± 0.004
w/o Subspace Protection	0.358 ± 0.003	2.478 ± 0.007	5.929 ± 0.062	0.086 ± 0.001	0.576 ± 0.004	0.793 ± 0.010
Unweighted Subspace	0.358 ± 0.005	2.440 ± 0.015	5.194 ± 0.060	0.086 ± 0.001	0.566 ± 0.010	0.794 ± 0.004
w/o Recovery	0.358 ± 0.004	2.472 ± 0.005	5.875 ± 0.032	0.085 ± 0.001	0.571 ± 0.005	0.799 ± 0.008

	WTH			ECL		
Variant	1	24	48	1	24	48
PAVE	$\mathbf{0.156 \pm 0.000}$	$\mathbf{0.425 \pm 0.001}$	$\mathbf{0.509 \pm 0.003}$	$\mathbf{2.301 \pm 0.045}$	$\mathbf{6.996 \pm 0.075}$	$\mathbf{8.207 \pm 0.096}$
w/o Online Correction	0.161 ± 0.001	0.438 ± 0.001	0.611 ± 0.006	2.319 ± 0.041	7.212 ± 0.035	8.453 ± 0.132
w/o Local Credit	0.157 ± 0.001	0.434 ± 0.005	0.613 ± 0.007	2.304 ± 0.045	7.095 ± 0.013	8.346 ± 0.140
w/o Version Awareness	0.157 ± 0.001	0.429 ± 0.001	0.599 ± 0.004	2.315 ± 0.035	7.067 ± 0.070	8.413 ± 0.309
w/o Direction Awareness	0.157 ± 0.000	0.438 ± 0.002	0.607 ± 0.007	2.307 ± 0.036	7.051 ± 0.047	8.293 ± 0.094
w/o Subspace Protection	0.159 ± 0.000	0.431 ± 0.005	0.601 ± 0.002	2.308 ± 0.039	7.045 ± 0.027	8.344 ± 0.073
Unweighted Subspace	0.158 ± 0.000	0.430 ± 0.003	0.590 ± 0.005	2.310 ± 0.025	7.066 ± 0.067	8.331 ± 0.133
w/o Recovery	0.158 ± 0.002	0.433 ± 0.001	0.601 ± 0.007	2.306 ± 0.041	7.100 ± 0.080	8.277 ± 0.079

also become increasingly important at longer horizons, consistent with the greater opportunity to exploit progressively revealed targets before the complete forecasting record matures. Finally, replacing responsibility-weighted Stable evidence with uniform weighting consistently reduces performance while preserving the remaining lifecycle, supporting the use of responsibility when constructing the protected predictive subspace.

E Sanity Check for High-Alignment Capability Retention

Equation 11 treats an expert as capability-retained when its representation alignment satisfies $a_{o,e} \geq \tau_a$, without additionally requiring its current loss to remain below the reference loss. We therefore perform a post-hoc diagnostic specifically on this high-alignment branch to measure how often predictive performance nevertheless degrades.

For each matured record, we identify its historical owner as

$$e_o^\star = \arg\max_e r_{o,e}, \tag{45}$$

consistent with the top-K_c credit assignment used by PAVE ($K_c = 1$). We restrict the analysis to historical owners satisfying $a_{o,e_o^\star} \geq \tau_a$; all such owners also satisfy the minimum responsibility threshold. For these cases, we define the relative capability degradation as

$$\Delta_{o,e_o^\star} = \frac{\bar{\ell}^{\mathrm{cur}}_{o,e_o^\star} - \bar{\ell}^{\mathrm{ref}}_{o,e_o^\star}}{\bar{\ell}^{\mathrm{ref}}_{o,e_o^\star} + \epsilon}. \tag{46}$$

This analysis is purely diagnostic and does not modify any routing, memory, or recovery decision.

Table 8 reports the conditional degradation statistics for the high-alignment subset in the retained diagnostic window, which contains randomly selected 10,000 retained high-alignment matured records. Among these cases, 95.8% exhibit no increase in current loss. Although 4.2% show some degradation, substantial degradation is considerably rarer: only 1.8% exceed a 10% relative loss increase, 0.84% exceed 20%, and 0.13% exceed 50%. These results indicate that, conditional on passing the alignment test, substantial predictive degradation is uncommon in the observed diagnostic window, while high representation alignment is not a strict functional guarantee.

F Supplementary Experiments

F.1 Controlled Recurring-Concept Drift

We further construct a controlled recurring-concept stream to examine whether the Recovery mechanism helps reacquire a previously useful capability after intervening online adaptation. The stream

Table 8: Conditional degradation analysis for the high-alignment branch of equation 11. We restrict the retained matured diagnostic records to historical owners satisfying $a_{o,e_o^\star} \geq \tau_a = 0.8$ and report the frequency of different levels of predictive degradation.

Performance change within high-alignment subset	Records	Fraction
No degradation ($\Delta \leq 0$)	9,580	95.80%
Any degradation ($\Delta > 0$)	420	4.20%
$\Delta > 5\%$	282	2.82%
$\Delta > 10\%$	180	1.80%
$\Delta > 20\%$	84	0.84%
$\Delta > 50\%$	13	0.13%

Table 9: Controlled $A_1 \to B \to A_2$ recurring-concept experiment. A_2 is generated from the same underlying regime as A_1. Results are mean $\pm$ standard deviation over three seeds.

Method	B-Early	B-Late	A_2-Early	Reacq. ↓	Excess Error ↓
w/o Recovery	0.2612 ± 0.0006	0.0594 ± 0.0007	0.2677 ± 0.0022	32.0 ± 1.0	3.59 ± 0.17
PAVE	0.2618 ± 0.0010	0.0595 ± 0.0004	$\mathbf{0.1796 \pm 0.0049}$	$\mathbf{19.0 \pm 1.7}$	$\mathbf{1.73 \pm 0.21}$

follows an $A_1 \to B \to A_2$ schedule, where A_2 uses exactly the same generating parameters as A_1. The transition from A_1 to B therefore requires adaptation to a genuinely new concept, whereas the return from B to A_2 tests whether a previously encountered capability can be reused after subsequent model updates.

Setup. The synthetic stream contains 900 timestamps and three variables, with A_1, B, and A_2 each occupying 300 timestamps. We use input length $L = 60$ and forecasting horizon $H = 24$, yielding forecast-origin intervals $[0, 240)$, $[240, 540)$, and $[540, 817)$, respectively. Regime A combines autoregressive dynamics, daily seasonality, a mild trend, and cross-variable mixing. Regime B changes the autoregressive dynamics, seasonal frequency and amplitude, trend direction, phase structure, and cross-variable mixing, producing a substantial distribution shift. Gaussian observation noise with standard deviation 0.05 is added throughout. Importantly, A_2 is an exact parameter copy of A_1 apart from newly sampled noise.

To keep the experiment controlled, we use a lightweight four-expert online mixture that follows the same causal release–update–predict ordering and version-aware Stable/Recovery lifecycle as PAVE. We compare full PAVE with *w/o Recovery*, which disables Recovery-memory admission and targeted replay while leaving the remaining online adaptation mechanisms unchanged. Results are averaged over three independent seeds.

Recovery metrics. We form a chronological error sequence by recording the first-horizon MSE across variables at each online origin. Let $\ell_{A_1}^{\mathrm{ref}}$ denote the mean MSE over the final 16 forecast origins of A_1. To characterize adaptation to the unseen intermediate concept, we report the mean MSE over the first eight and final 16 origins of B, denoted B-Early and B-Late, respectively. We similarly report the mean error over the first eight origins of the recurring regime as A_2-Early.

Reacquisition time measures how quickly the model returns to its previous A-regime performance after A_2 begins. Specifically, it is the first offset in A_2 at which the four-origin rolling MSE falls below $1.2\,\ell_{A_1}^{\mathrm{ref}}$ and remains below this threshold for four consecutive positions. *Cumulative excess error* sums $\max(\ell_t - \ell_{A_1}^{\mathrm{ref}}, 0)$ throughout A_2.

Results. The two variants respond almost identically when the stream first changes from A_1 to the unseen regime B. Their early-B MSE rises sharply to about 0.262, compared with an A_1 reference error of about 0.066, showing that the transition introduces a substantial new-concept mismatch. Both variants then adapt along similar trajectories and reach nearly identical late-B performance (0.0595 for PAVE and 0.0594 without Recovery). Thus, Recovery does not provide a noticeable advantage when adapting to the currently active, previously unseen concept.

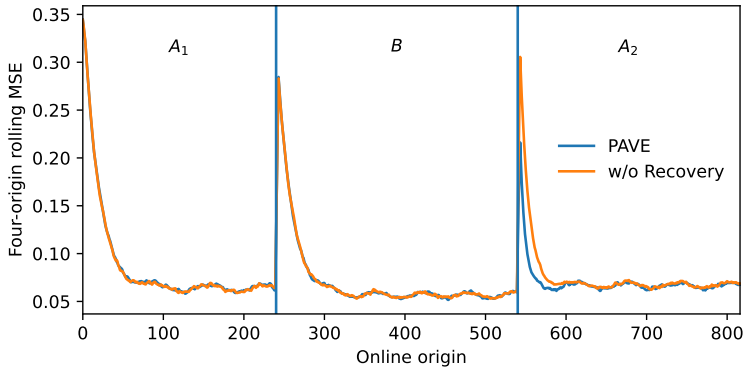


Figure 4: Four-origin rolling MSE on the controlled $A_1 \to B \to A_2$ stream. Both variants experience a similar error spike when the unseen regime B first appears and adapt to it at comparable rates. When the previously observed regime A returns, PAVE shows a smaller recurrence spike and faster reacquisition than w/o Recovery.

The behavior differs when the previously observed regime A returns. Both variants experience another error increase at the beginning of A_2, because their current parameters have adapted to B, but the increase is substantially smaller for PAVE. Recovery reduces the early-A_2 MSE from 0.2677 to 0.1796, a 32.9% reduction. It also decreases the mean reacquisition time from 32.0 to 19.0 online origins, corresponding to a 40.6% shorter recovery period, and reduces cumulative excess error from 3.59 to 1.73, a 51.8% reduction.

Figure 4 highlights the different roles of the two transitions. At $A_1 \to B$, PAVE and w/o Recovery exhibit a comparable error spike and subsequently adapt to the new regime at similar rates. At $B \to A_2$, PAVE exhibits a smaller recurrence spike and returns to its previous A-regime error level more rapidly. Because A_2 repeats the generating mechanism of A_1, this asymmetry isolates the intended role of Recovery: it provides little benefit for learning a genuinely unseen concept, but facilitates faster reuse of a previously useful capability after that capability has been affected by intervening online updates.

F.2 Capacity-Matched Controls and Computational Cost

We further examine whether PAVE's improvement can be explained primarily by the capacity of its four-expert architecture or by a larger online computational budget. To isolate these factors, we construct two controls that retain the same heterogeneous expert pool as PAVE.

Capacity-matched controls. *Matched MoE* uses the same four experts (two FSNet and two FSNet-Time), neural router, initialization, progressive-feedback routing, and online optimization schedule as PAVE, but removes version-aware validation, Stable/Recovery memory management, subspace protection, and targeted Recovery. To match the amount of historical supervision used during online updates, whenever PAVE performs Recovery replay, Matched MoE instead samples the same number of matured records uniformly from a FIFO buffer. Thus, the two methods have identical forecasting capacity and a matched replay/update budget, while differing in how historical evidence is validated, selected, and reused.

Uniform MoE instantiates the same four experts but fixes the mixture weights to $p_e = 1/E$ throughout online evaluation. It does not learn the neural router and does not use version-aware memory. This provides a simpler control for testing whether the gain can be explained by merely combining multiple heterogeneous forecasters.

The capacity-matched results show a clear horizon-dependent separation between architectural capacity and evidence management. At $H = 1$, where feedback matures quickly, PAVE improves over Matched MoE by only 1.9% on average across datasets. The difference increases to 4.4% at $H = 24$ and 15.4% at $H = 48$. Relative to Uniform MoE, the corresponding reductions are 5.7%, 7.4%,

Table 10: Single-seed capacity-matched comparison in MSE (lower is better). Matched MoE retains PAVE's four-expert architecture, neural router, and online update/replay budget but removes the version-aware evidence lifecycle. Uniform MoE uses the same four experts with fixed uniform routing.

	ETTh2			ETTm1			WTH			ECL		
Method	1	24	48	1	24	48	1	24	48	1	24	48
Uniform MoE	0.377	2.680	6.950	0.092	0.602	0.881	0.165	0.453	0.647	2.398	7.454	9.057
Matched MoE	0.363	2.655	6.904	0.087	0.572	0.836	0.159	0.437	0.612	2.331	7.242	8.710
PAVE	**0.355**	**2.403**	**4.906**	**0.085**	**0.554**	**0.760**	**0.157**	**0.426**	**0.507**	**2.286**	**7.070**	**8.149**

Table 11: Computational cost on ECL-48 using a single NVIDIA A100-SXM4-80GB GPU. Total parameters include all instantiated model parameters; online parameters denote parameters optimized by gradient updates during online adaptation. Runtime covers the complete online-origin cycle with batch size one. Persistent state denotes auxiliary online storage beyond model parameters.

Method	Total Params. (M)	Online Params. (M)	Time / Origin (ms)	Peak GPU Mem. (GB)	Persistent State (MB)
DynaME	1.369	0.052	30	0.05	32.6
PatchTST-DGrad	3.493	3.493	18	0.10	6.7
PROCEED	3.635	0.142	16	0.06	24.7
ER	3.493	3.493	45	0.37	24.7
DSOF	3.495	3.495	41	0.35	24.8
FSNet	6.264	5.982	27	0.15	24.7
OneNet	7.584	7.018	52	0.20	24.7
Uniform MoE	16.565	14.032	70	0.62	6.7
Matched MoE	16.565	15.434	125	0.71	69.1
PAVE	16.565	15.434	165	0.75	71.6

and 18.7%. Therefore, simply instantiating the same four experts, or even combining them with a learned router and matched replay budget, does not reproduce PAVE's long-horizon performance.

This trend is consistent with the delayed-evidence motivation of PAVE. As the forecasting horizon grows, a prediction remains unresolved for more online updates before its complete target becomes available, increasing the mismatch between historical expert ownership and current expert capability. The growing gap between PAVE and the capacity-matched controls therefore supports the role of version-aware validation and structured evidence reuse rather than model capacity alone.

Computational cost. We additionally compare the online computational cost on ECL-48, the largest-variable and longest-horizon setting among the evaluated benchmarks. All methods use batch size one and a single NVIDIA A100-SXM4-80GB GPU. Runtime covers the complete online-origin cycle, including processing all causally available feedback, online adaptation, and generation of the next forecast, rather than prediction alone. Parameter counts correspond to the instantiated ECL-48 models. Peak GPU memory is measured using PyTorch allocated memory, while persistent state accounts for online buffers, historical evidence, and other retained state beyond model parameters, excluding the dataset and evaluation-only diagnostics.

As expected, the four-expert variants use more parameters than the single-model baselines because all heterogeneous experts remain instantiated throughout online forecasting. This additional capacity, however, is controlled directly by Matched MoE: PAVE and Matched MoE have exactly the same total model size (16.565M parameters) and the same number of gradient-optimized online parameters (15.434M). Their accuracy difference therefore cannot be attributed to additional forecasting parameters.

The version-aware evidence lifecycle primarily introduces runtime rather than model-size overhead. Relative to Matched MoE, PAVE increases the per-origin runtime from 125 to 165 ms, while peak GPU memory increases only from 0.71 to 0.75 GB and persistent online state from 69.1 to 71.6 MB. The additional cost comes from capability re-evaluation, periodic memory reassessment, Stable-subspace construction, and targeted Recovery replay. These mechanisms add no additional forecasting experts or prediction parameters. Overall, the capacity-matched comparison indicates that PAVE's long-horizon gains arise from how delayed historical evidence is managed, rather than from increasing the underlying expert capacity.