MOST-RECENT ANCHORING WITH RECURRENT ORDERING FOR TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term forecasting models commonly process all patches in a look-back window using the same fixed stack. Older contextual patches and recent evidence therefore receive the same computational depth. Yet the information closest to the forecast and the more distant context do not contribute equally. Uniform processing leaves this distinction unexpressed in the architecture. We propose MARO, a **M**ost-Recent **A**nchoring with **R**ecurrent **O**rdering model that processes the look-back window from the most recent patch to the oldest. The most recent patch serves as the anchor. It initializes the latent state and conditions each subsequent step, so older patches are folded into a representation that remains centered on recent evidence. A single shared module is reused at every step, so extending the scan further into the past introduces no additional parameters. Intermediate states retained during the scan allow the forecast head to weigh short and long portions of the history separately. This expresses recency through the order of recurrent refinement. Extensive experiments across multiple real-world time series datasets show that MARO achieves state-of-the-art performance on both long-term and short-term forecasting tasks. Ablation studies examine the contribution of the main architectural components.

1 INTRODUCTION

Time series forecasting (TSF) supports decision-making in domains such as energy and climate as well as transportation (Max Planck Institute for Biogeochemistry, 2024; Trindade, 2015; California Department of Transportation (Caltrans), 2024). Long-term TSF has been studied with attention-based (Liu et al., 2024) and convolutional (Liu et al., 2022) architectures as well as recurrent (Lin et al., 2026) and MLP-based architectures (Chen et al., 2023; Wang et al., 2024). Many non-recurrent models map the look-back window through a predetermined stack of processing blocks. Patch-based models first divide the window into tokens and then apply the same fixed-depth processing to every patch (Nie et al., 2023; Ekambaram et al., 2023). Each patch undergoes the same number of transformations regardless of its distance from the forecast origin.

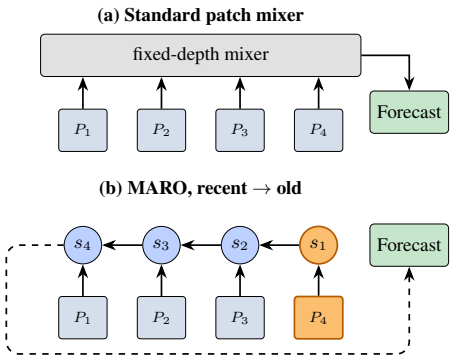


Figure 1: Conventional patch mixers transform every patch at the same fixed depth in panel (a). MARO starts from the most recent patch and recurrently incorporates older patches in panel (b).

Forecasting uses the positions in a look-back window asymmetrically. Patches adjacent to the forecast boundary often provide the most immediate evidence. Earlier patches more often supply longer-range context. This observation motivates anchoring the forecast in the most recent patch and incorporating progressively older patches as contextual corrections. Positional information can encode recency, but it does not give patches different numbers of recurrent refinement steps. A fixed stack still subjects every patch to the same number of transformations. We use computational depth to mean the number of recurrent refinement steps through which the information from a patch is carried.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

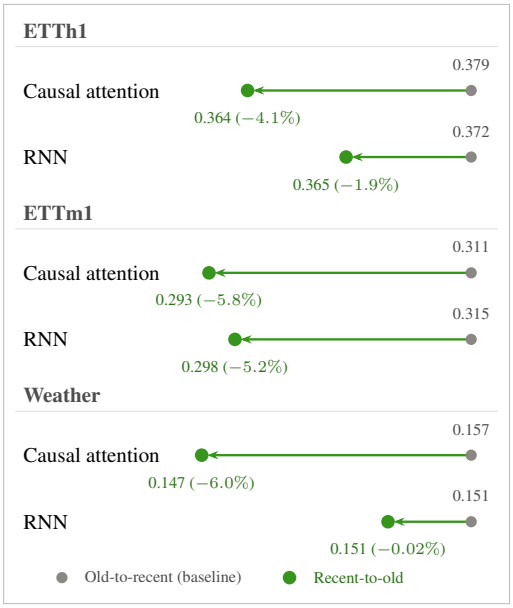


Figure 2: Test MSE under old-to-recent and recent-to-old scanning on ETTh1, ETTm1, and Weather. Values are truncated to three decimals; percentage changes use the original values. Arrow lengths are schematic.

To test whether reversing the scan provides a useful recency bias across update mechanisms, we compared recent-to-old and old-to-recent ordering with causal attention and RNN updates. Recent-to-old begins at the forecast boundary and moves backward, whereas old-to-recent begins with the earliest patch and moves forward. Under causal attention, this reversal places recent context in the visible prefix. Under the RNN update, it initializes the hidden state from the most recent patch before adding older history. Recent-to-old scanning produced lower test MSE in all six comparisons on ETTh1, ETTm1, and Weather in Figure 2. For causal attention, the reductions are 4.1, 5.8, and 6.0 percent, respectively; for RNN updates, they are 1.9, 5.2, and 0.02 percent. The Weather RNN difference is smaller than the three-decimal display precision. Appendix D.1 reports the MSE and MAE for all five datasets. The consistency across the two mechanisms provides empirical evidence that the benefit of reversing the scan to recent-to-old is not specific to one update rule.

Recurrent-depth computation allows this effective depth to vary. Recent reasoning systems introduce intermediate computation through generated reasoning tokens or continuous latent states (Wei et al., 2022; Hao et al., 2025), while recurrent-depth models explicitly reuse a shared block to increase latent computation without increasing the parameter count (Geiping et al., 2025). Applied to a look-back window, the computation assigned to a patch can depend on its temporal position. Fixed-depth patch mixers do not express this depth asymmetry.

We propose MARO, a **M**ost-Recent **A**nchoring with **R**ecurrent **O**rdering model that scans overlapping patches from the most recent patch to the oldest (Figure 1). The most recent patch initializes the latent state and serves as a fixed anchor while a shared module progressively integrates earlier patches as context. MARO retains intermediate states along this scan and combines them for forecasting, allowing the prediction to draw separately on shorter- and longer-range portions of the history. The same module is reused at every step, so incorporating older patches does not add parameters.

Our contributions are threefold.

- We identify uniform computational depth as a structural limitation of patch-based forecasting and formulate the history as a sequence of recurrent refinements organized by recency.
- We introduce MARO as a parameter-shared architecture that anchors the forecast at the most recent patch and progressively integrates older context. This design makes recency explicit in the computation while keeping the recurrent parameters independent of the number of patches.
- We evaluate MARO across long-term and short-term forecasting benchmarks. The results and controlled ablations show when recency-aware computation helps and how each architectural component contributes to forecasting accuracy. The code is available at `https://anonymous.4open.science/r/MARO-7B20/README.md`.

2 Related Work

Time Series Forecasting. Recent forecasting baselines span linear mappings and convolutions as well as attention and multiscale mixing. DLinear and RLinear use direct linear mappings (Zeng et al., 2023; Li et al., 2023). SCINet models temporal dynamics through convolution and interac-

tion (Liu et al., 2022). iTransformer represents variables as tokens (Liu et al., 2024). PatchTST and Crossformer organize the input into temporal patches (Nie et al., 2023; Zhang & Yan, 2023). TimeMixer and TimeKAN learn multiscale or frequency-decomposed representations (Wang et al., 2024; Huang et al., 2025). Amplifier emphasizes low-energy components (Fei et al., 2025). CycleNet models periodic structure (Lin et al., 2024). SRSNet selects among alternative patch representations (Wu et al., 2025). These models form the principal comparison set in our long-term and short-term experiments. Their architectures differ, but they process the input through stacks of predetermined depth.

Patching. PatchTST divides time series into overlapping temporal segments (Nie et al., 2023). Crossformer additionally captures dependencies between variables (Zhang & Yan, 2023). Subsequent approaches introduce length-extendable patching to alleviate information loss at fixed patch boundaries (Huang et al., 2024). Other methods choose among alternative patch representations (Wu et al., 2025) or integrate patching with wavelet analysis (Murad et al., 2025). Some build and prune graphs among patches (Hu et al., 2025). Each patch in these approaches is generally embedded once before all patch representations are processed together. MARO instead right-aligns overlapping patches with any padding placed on the oldest side and incorporates their tokens sequentially through a recurrent scan.

Recurrent Depth. Recent language models reason through continuous latent representations (Hao et al., 2025) or iteratively refine hidden states through recurrent depth (Geiping et al., 2025). MARO draws on shared recurrent computation but assigns a distinct historical role to each step. Each recurrence introduces one additional older patch instead of processing an unchanged token set again.

Recurrent processing is also well established in time series forecasting. SegRNN reduces recurrent iterations through segment-wise GRU processing and parallel multi-step forecasting (Lin et al., 2026). xLSTMTime employs xLSTM cells (Alharthi & Mahmood, 2024). S-Mamba applies bidirectional Mamba over variate representations to model inter-variate correlations, followed by a feed-forward network for temporal dependencies (Wang et al., 2025). MARO instead traverses temporal patches from the forecast boundary to the oldest patch, retaining the most recent patch as a persistent anchor. Each subsequent step introduces one older patch, so its traversal order encodes recency.

3 METHOD

3.1 PROBLEM SETUP AND OVERVIEW

Let $\mathbf{X} \in \mathbb{R}^{C \times L}$ be the C-channel look-back window and $\mathbf{Y} \in \mathbb{R}^{C \times H}$ the target. Let $\mathbf{M}^{\text{hist}}$ and $\mathbf{M}^{\text{fut}}$ denote timestamp-derived covariates aligned with the look-back and forecast horizon, respectively. They are known from the timestamps; Appendix B.1 describes their dataset-specific construction and cyclic augmentation.

Figure 3 illustrates the overall MARO architecture. The normalized window is right-aligned into patches (Section 3.2). Embedding, calendar conditioning, and sparse channel mixing yield tokens $U_{c,1}, \ldots, U_{c,N}$, with $U_{c,N}$ at the forecast boundary (Section 3.3). The recurrent module then operates on each channel independently with shared weights. We suppress c and write its tokens as $U_1, \ldots, U_N$. It initializes from U_N and folds in $U_{N-1}, \ldots, U_1$ while maintaining a temporal context and a forecast latent (Section 3.4). Exponentially spaced latent states are fused into $\bar{z}$ (Section 3.5). An MLP decodes the fused state and adds raw-input and future-calendar residuals (Section 3.6).

3.2 OVERLAPPING PATCHING

MARO first applies reversible instance normalization (RevIN) to the look-back window, yielding $\widetilde{\mathbf{X}} = \mathrm{RevIN}(\mathbf{X}) \in \mathbb{R}^{C \times L}$ (Kim et al., 2022). For channel c, let $\mathbf{p}_{c,n} \in \mathbb{R}^P$ be the n-th right-aligned patch of $\widetilde{\mathbf{X}}_c$, with length P, stride S, and $N = \lceil \max(0, L - P)/S \rceil + 1$. Padding is placed only on the oldest side, so $\mathbf{p}_{c,N}$ ends exactly at the forecast boundary.

Within each patch, an intra-patch MLP mixes its P samples through the residual transform $\mathbf{p}_{c,n} + \mathrm{MLP}_{\text{intra}}(\mathbf{p}_{c,n})$. Projection to width d and a residual token MLP then produce the embedded token $T_{c,n} \in \mathbb{R}^d$.

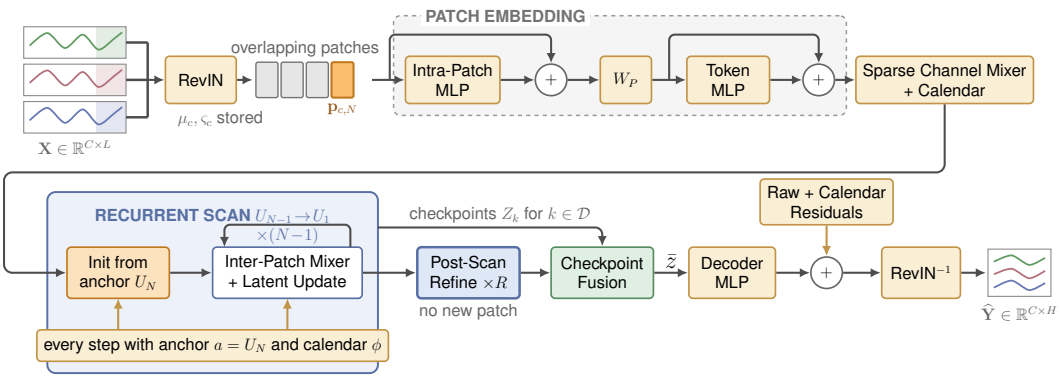


Figure 3: Overview of the MARO architecture.

3.3 Sparse Channel Mixing and Calendar Conditioning

For each target channel c, MARO scores the other channels using learned channel embeddings, retains the κ highest-scoring neighbours, and mixes their patch tokens with the target token using normalized weights. A bounded gate, initialized near zero, controls the resulting residual correction. Setting $\kappa = 0$ disables cross-channel exchange, whereas $\kappa = C - 1$ recovers dense mixing.

The scorer is shared across patches and depends on learned channel identities, not on the current input window. One sparse neighbourhood is therefore used for every sample and patch, avoiding the dense per-sample $C \times C$ cross-channel attention over variate tokens used in iTransformer (Liu et al., 2024). The selected neighbour indices are discrete, but their scores retain gradients, so the routing can change during training. We interpret the learned edges as routing choices, not as recovered correlation structure. The complete scoring and gated-update formulation is given in Appendix E.1. Topology controls are reported in Appendix E.2.

For each historical patch n, an MLP encodes the boundary and midpoint features from $\mathbf{M}^{\text{hist}}$ into calendar token b_n, added identically to each embedded token $T_{c,n}$ before sparse mixing. Because the mixing weights sum to one, this shared token appears in both the weighted neighbourhood token and the target token and therefore cancels from the residual difference used for channel correction. The resulting token $U_{c,n} \in \mathbb{R}^d$ retains calendar information after sparse mixing.

A second MLP encodes $\mathbf{M}^{\text{fut}}$ into per-step tokens $f_t \in \mathbb{R}^d$. Selected boundary and midpoint tokens are compressed into the fixed future-context vector $\phi \in \mathbb{R}^d$ used by every recurrent update. The full sequence $\{f_t\}_{t=1}^H$ is retained for the calendar residual in Section 3.6. For each channel, $U_1, \ldots, U_N$ are the tokens passed to the scan; recurrence is channel-wise with shared weights. The neighbourhood fraction $\delta = \kappa/(C-1)$ is reported for every setting in Appendix C.2.

3.4 Recurrent Latent Refinement

Figure 4 shows the shared recurrent module called the *scan*. It starts from U_N and processes each older patch once in the order $U_{N-1}, \ldots, U_1$. At scan step s, the incoming older token is U_{N-s}, the $(s+1)$-th most recent token. The scan runs independently for each channel. We suppress the channel index and take all vectors below to lie in $\mathbb{R}^d$.

Context and latent state. At scan step s, the context state $h^{(s)}$ summarizes the patches $U_N, \ldots, U_{N-s}$ incorporated so far without using horizon information. The corresponding forecast latent $z^{(s)}$ is updated from that context, the recent anchor, and the future calendar. Separating the two states keeps temporal comparison in $h^{(s)}$ distinct from horizon-conditioned refinement in $z^{(s)}$.

Initialization. The fixed anchor $a = U_N$ initializes both states.

$$h^{(0)} = a, \qquad z^{(0)} = a + \text{MLP}_{\text{init}}\big([\,a \,\|\, \phi\,]\big), \tag{1}$$

where $\|$ denotes feature concatenation. The latent begins at the forecast boundary after adjustment for the future calendar.

Positional embedding. For $s = 1, \ldots, N-1$, the normalized scan position is encoded as

$$\mathrm{pos}_s = \mathrm{MLP}_{\mathrm{step}}\left(\frac{s}{N-1},\ \left(\frac{s}{N-1}\right)^2,\ \sin\frac{2\pi s}{N-1},\ \cos\frac{2\pi s}{N-1}\right). \tag{2}$$

This allows the same encoder to handle different scan lengths.

Progressive temporal mixer. The inter-patch MLP compares the incoming older patch with the accumulated recent context.

$$\rho_s = \mathrm{MLP}_{\mathrm{rel}}\Big(\big[\, h^{(s-1)} \,\|\, U_{N-s} \,\|\, U_{N-s} \odot h^{(s-1)} \,\|\, \mathrm{pos}_s \,\big]\Big). \tag{3}$$

The elementwise product $U_{N-s} \odot h^{(s-1)}$ is a multiplicative interaction between the incoming patch and the accumulated context. Reusing the same mixer at every step extends this pairwise comparison over the complete temporal window without adding parameters with scan depth.

Gated additive updates. A shared bounded gate $g_r = \bar{g}_r \sigma(\ell_r)$ controls both state updates. Here $\ell_r \in \mathbb{R}$ is a learned scalar logit and $\bar{g}_r$ a fixed upper bound, giving $0 < g_r < \bar{g}_r$. The same scalar is reused at every scan step for the context and latent residual updates.

$$h^{(s)} = h^{(s-1)} + g_r\, \rho_s, \tag{4}$$

$$z^{(s)} = z^{(s-1)} + g_r \underbrace{\mathrm{MLP}_{\mathrm{upd}}\Big(\big[\, z^{(s-1)} \,\|\, h^{(s)} \,\|\, \rho_s \,\|\, a \,\|\, \phi \,\|\, \mathrm{pos}_s \,\big]\Big)}_{u_s}. \tag{5}$$

The pre-gate latent-update vector u_s is the output of $\mathrm{MLP}_{\mathrm{upd}}$ at step s; its actual additive increment is $g_r u_s$. This update uses the new context, temporal relation, fixed anchor, and future calendar.

The relation ρ_s reaches the latent through the accumulated context and a direct input. These paths provide the temporal summary and the current patch relation, respectively. The anchor a and future context ϕ remain fixed throughout the scan.

3.5 Checkpointed Multi-Depth Fusion

MARO retains checkpoint states at exponentially spaced temporal depths and aggregates them through trainable fusion weights. For each channel, suppressing c, $Z_k := z^{(k-1)}$ is the state after its k most recent patches have been consumed. The checkpoint depths are

$$\mathcal{D} = \{2^j : 0 \le j \le \lfloor \log_2 N \rfloor\} \cup \{N\}, \qquad |\mathcal{D}| = O(\log N). \tag{6}$$

This concentrates resolution near the forecast boundary while using only $O(\log N)$ states. Shared softmax weights are initialized uniformly and combine them without increasing the latent width.

$$\alpha = \mathrm{softmax}\big(\ell^{\mathrm{fuse}}\big) \in \mathbb{R}^{|\mathcal{D}|}, \qquad \bar{z}_c = \sum_{k \in \mathcal{D}} \alpha_k\, Z_k \quad \text{for every channel } c. \tag{7}$$

$\bar{z}_c \in \mathbb{R}^d$ regardless of N. The same α is used for every window and channel within a trained configuration. At depth N, the stored checkpoint is replaced by the post-scan refined state before this fusion is evaluated (Section 3.6).

Direct weighting of recurrent updates.

Proposition 1 (Monotone direct weighting of recurrent updates)**.** *Conditioned on the realized scan and post-scan update vectors, checkpoint fusion assigns non-increasing direct coefficients to the recurrent updates generated along the recent-to-old scan. Here u_s is the output of $\mathrm{MLP}_{\mathrm{upd}}$ in equation 5 before multiplication by g_r, and v_r is the output of the same MLP at post-scan refinement*

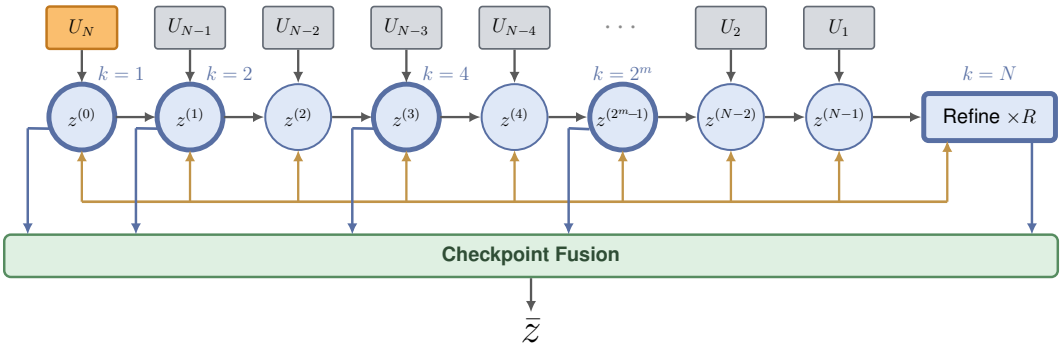


Figure 4: Recurrent processing from the most recent patch to the oldest. The anchor and future context condition every update. Selected intermediate states and the refined final state are fused into $\bar{z}$.

step r in equation 9, also before multiplication by g_r. The actual additions to the latent are $g_r u_s$ and $g_r v_r$, respectively. With the step-independent recurrent gate g_r,

$$\bar{z} = z^{(0)} + g_r \sum_{s=1}^{N-1} \lambda_s \, u_s + \alpha_N g_r \sum_{r=1}^{R} v_r, \qquad \lambda_s = \sum_{k \in \mathcal{D},\, k>s} \alpha_k, \tag{8}$$

where α is the softmax weight vector over checkpoint depths $\mathcal{D}$. For $N \geq 2$, its non-negative entries give $\lambda_1 \geq \cdots \geq \lambda_{N-1} \geq 0$, with $\lambda_1 = 1 - \alpha_1$ and $\lambda_{N-1} = \alpha_N$. The direct coefficient multiplying $z^{(0)}$ is identically one because $\sum_{k \in \mathcal{D}} \alpha_k = 1$.

Appendix F.1 provides the proof. This result concerns the direct additive coefficients assigned by checkpoint fusion to the realized recurrent update vectors. It does not imply that the end-to-end sensitivity of the forecast to individual raw input patches is monotone, because the update vectors are nonlinear functions of recursively accumulated states. $z^{(0)}$ itself contains both the anchor and the initialization MLP, and the anchor also conditions later recurrent updates; its direct coefficient does not describe the total end-to-end influence of the raw anchor.

3.6 Post-Scan Refinement and Forecast Decoding

During the recurrent scan, each latent update incorporates a newly arriving older patch while adjusting the forecast latent. The complete historical summary $h^{(N-1)}$ becomes available only after the final patch has been incorporated. Decoding without further refinement would provide no latent-update step with that summary held fixed. We therefore reuse $\mathrm{MLP}_{\mathrm{upd}}$ for R additional steps after the scan. These steps receive no new historical patch: $h^{(N-1)}$, the recent anchor a, future context ϕ, and terminal position $\mathrm{pos}_{\max(1,N-1)}$ remain fixed, while the relation-input slot is set to $\mathbf{0} \in \mathbb{R}^d$. This separates evidence acquisition from forecast-state refinement and adds latent computational depth without a separate recurrent block or step-specific parameters, following the shared-module motivation of recurrent-depth models (Geiping et al., 2025).

$$z \;\leftarrow\; z + g_r \, \mathrm{MLP}_{\mathrm{upd}}\big([\, z \,\|\, h^{(N-1)} \,\|\, \mathbf{0} \,\|\, a \,\|\, \phi \,\|\, \mathrm{pos}_{\max(1,N-1)} \,]\big). \tag{9}$$

This use of recurrent refinement is related in spirit to xLSTM-Mixer (Kraus et al., 2025), which first constructs a shared linear forecast and subsequently refines it through xLSTM blocks. MARO differs in both the object and timing of refinement: it refines a latent forecast state only after the recent-to-old scan has aggregated the complete look-back window, and it reuses the same $\mathrm{MLP}_{\mathrm{upd}}$ employed during scanning. The refined state replaces the deepest checkpoint before fusion. Setting $R = 0$ omits post-scan refinement: the unrefined deepest state remains the depth-N checkpoint in the usual fusion before decoding. The latent decoder gives $\widehat{\mathbf{Y}}^{\mathrm{lat}}_{c,:} = \mathrm{MLP}_{\mathrm{dec}}(\bar{z}_c)$; zero-initialized residual branches use the normalized input and future calendar.

$$\widehat{\mathbf{Y}}^{\mathrm{norm}} = \widehat{\mathbf{Y}}^{\mathrm{raw}} + \widehat{\mathbf{Y}}^{\mathrm{lat}} + \widehat{\mathbf{Y}}^{\mathrm{cal}}, \tag{10}$$

Here $\widehat{\mathbf{Y}}^{\mathrm{raw}}_{c,:} = W_{\mathrm{raw}}\widetilde{\mathbf{X}}_c$ is the raw-input residual and $\widehat{Y}^{\mathrm{cal}}_{c,t} = v_c^\top f_t + b_c$ is the future-calendar residual, with learned channel-specific v_c, b_c. The final forecast is $\widehat{\mathbf{Y}} = \mathrm{RevIN}^{-1}(\widehat{\mathbf{Y}}^{\mathrm{norm}}) \in \mathbb{R}^{C\times H}$.

4 EXPERIMENTS

Protocol. The long-term suite comprises ETTh1, ETTh2, ETTm1, and ETTm2 (Zhou et al., 2021), Weather (Max Planck Institute for Biogeochemistry, 2024), Electricity (Trindade, 2015), Traffic (California Department of Transportation (Caltrans), 2024), and Solar (National Renewable Energy Laboratory, 2024). The short-term suite comprises PEMS03, PEMS04, PEMS07, and PEMS08 (California Department of Transportation (Caltrans), 2024). The long-term horizons are $H \in \{96, 192, 336, 720\}$, and the PEMS horizons are $H \in \{12, 24, 48, 96\}$.

The long-term baselines are SRSNet (Wu et al., 2025), TimeKAN (Huang et al., 2025), Amplifier (Fei et al., 2025), iTransformer (Liu et al., 2024), TimeMixer (Wang et al., 2024), and PatchTST (Nie et al., 2023). The short-term baselines are CycleNet/MLP (Lin et al., 2024), iTransformer (Liu et al., 2024), SCINet (Liu et al., 2022), and PatchTST (Nie et al., 2023). The efficiency comparison also includes DLinear (Zeng et al., 2023). Long-term baseline results are quoted from the published SRSNet comparison (Wu et al., 2025), which uses the same long-term look-back candidates $\{96, 336, 512\}$. We do not independently retune these baselines; short-term baseline results are quoted from their published sources.

A fixed look-back length rewards whichever method that length happens to suit (Abdelmalak et al., 2026). Following SRSNet, we select the look-back length for each long-term dataset and horizon from $\{96, 336, 512\}$. PEMS uses a fixed 96-step look-back. MARO is trained with mean squared error. The best value is bold and the second best is underlined. Appendix B describes the datasets and evaluation protocol. Appendix C gives the search settings, and Appendix D reports the full long-term results.

4.1 MAIN RESULTS

Table 1: Long-term results averaged over $H \in \{96, 192, 336, 720\}$, with the four ETT subsets merged into one group. Full per-dataset, per-horizon results and standard deviations are reported in Appendix D.

	MARO (Ours)		SRSNet		TimeKAN		Amplifier		iTransformer		TimeMixer		PatchTST	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETT (Avg)	**0.329**	**0.372**	<u>0.335</u>	<u>0.375</u>	0.340	0.380	0.346	0.383	0.351	0.384	0.346	0.383	0.343	0.381
Weather	**0.223**	**0.261**	<u>0.225</u>	0.265	0.226	0.269	**0.223**	0.263	0.230	0.269	<u>0.225</u>	<u>0.262</u>	**0.223**	<u>0.262</u>
Electricity	**0.155**	**0.252**	<u>0.160</u>	<u>0.254</u>	0.163	0.257	0.162	0.255	0.162	0.258	0.184	0.281	0.170	0.270
Traffic	**0.372**	**0.265**	<u>0.391</u>	<u>0.270</u>	0.419	0.285	0.412	0.290	0.397	0.280	0.409	0.278	0.396	0.275
Solar	<u>0.186</u>	<u>0.245</u>	**0.183**	**0.238**	0.198	0.263	0.202	0.281	0.201	0.260	0.203	0.260	0.200	0.284
# Best / Total	**4/5**	**4/5**	<u>1/5</u>	<u>1/5</u>	0/5	0/5	<u>1/5</u>	0/5	0/5	0/5	0/5	0/5	<u>1/5</u>	0/5

Long-term forecasting. Table 1 averages the four horizons of each dataset and merges the four ETT subsets into one row. MARO achieves the best or tied best result on four of the five groups. The largest gains occur on Traffic and Electricity, where MARO improves over the strongest baseline by about five and three percent in MSE, respectively. On the combined ETT group, it lowers MSE from 0.335 to 0.329 while retaining the best MAE. On Weather, MARO ties the best MSE and attains the lowest MAE. Solar is the only group it does not lead. Its MSE and MAE trail SRSNet by 0.003 and 0.007. Appendix D provides the disaggregated results for every dataset and horizon.

Short-term forecasting. The PEMS comparison follows the baseline set and 96-step-input protocol reported by CycleNet (Lin et al., 2024). MARO ranks first in 8 of 16 MSE cells and 6 of 16 MAE cells. The next-highest counts are 4 and 5, respectively. It leads both metrics at $H \in \{12, 24\}$ on PEMS03, PEMS07, and PEMS08, and it also retains the best MSE on PEMS08 at $H \in \{48, 96\}$. PEMS04 is the clear exception, where SCINet leads all four horizons. At longer horizons, iTransformer and SCINet become more competitive.

Table 2: Full PEMS forecasting results. Baseline values are quoted from the published CycleNet comparison.

Dataset	H	MARO (Ours) MSE	MARO (Ours) MAE	CycleNet/MLP MSE	CycleNet/MLP MAE	iTransformer MSE	iTransformer MAE	SCINet MSE	SCINet MAE	PatchTST MSE	PatchTST MAE
PEMS03	12	**0.064**	**0.168**	0.066	0.172	0.071	0.174	0.066	0.172	0.099	0.216
	24	**0.083**	**0.190**	0.089	0.201	0.093	0.201	0.085	0.198	0.142	0.259
	48	0.131	0.239	0.136	0.247	**0.125**	**0.236**	0.127	0.238	0.211	0.319
	96	0.188	0.298	0.182	0.282	**0.164**	**0.275**	0.178	0.287	0.269	0.370
PEMS04	12	0.075	0.179	0.078	0.186	0.078	0.183	**0.073**	**0.177**	0.105	0.224
	24	0.094	0.204	0.099	0.212	0.095	0.205	**0.084**	**0.193**	0.153	0.275
	48	0.136	0.250	0.133	0.248	0.120	0.233	**0.099**	**0.211**	0.229	0.339
	96	0.196	0.309	0.167	0.281	0.150	0.262	**0.114**	**0.227**	0.291	0.389
PEMS07	12	**0.055**	**0.148**	0.062	0.162	0.067	0.165	0.068	0.171	0.095	0.207
	24	**0.081**	**0.183**	0.086	0.192	0.088	0.190	0.119	0.225	0.150	0.262
	48	0.128	0.238	0.128	0.234	**0.110**	**0.215**	0.149	0.237	0.253	0.340
	96	0.148	0.257	0.176	0.268	**0.139**	0.245	0.141	**0.234**	0.346	0.404
PEMS08	12	**0.075**	**0.176**	0.082	0.185	0.079	0.182	0.087	0.184	0.168	0.232
	24	**0.100**	**0.197**	0.117	0.226	0.115	0.219	0.122	0.221	0.224	0.281
	48	**0.152**	0.245	0.169	0.268	0.186	**0.235**	0.189	0.270	0.321	0.354
	96	**0.202**	0.278	0.233	0.306	0.221	**0.267**	0.236	0.300	0.408	0.417
# Best / Total		**8/16**	**6/16**	0/16	0/16	4/16	5/16	4/16	5/16	0/16	0/16

4.2 Ablation Study

Table 3 compares MARO with variants that remove its calendar input and channel mixer as well as its temporal mixing stages. It also includes an anchor-only predictor and a *full-window MLP* that replaces the recurrent scan and fusion with one MLP over all patch tokens.

The complete model performs best on nearly every dataset and metric. The most consistent degradation appears when temporal processing is weakened. Decoding only the anchor discards useful history while the full-window MLP loses the progressive integration and multi-depth fusion of MARO. Removing refinement and processing the full window at once also reduce accuracy across all datasets. Channel mixing is particularly important on ETTh2 but has a smaller effect on Weather and Electricity. Calendar conditioning generally helps, while its effect on Weather is negligible. The topology controls in Appendix E.2 separately evaluate whether the selected channel graph must be learned.

Table 3: MARO component ablations averaged over $H \in \{96, 192, 336, 720\}$. Values are truncated to three decimal places. The best value in each column is bold and the second best is underlined. The full model uses the same per-horizon repeated-seed means as the main results; each ablation uses one validation-selected seed.

Variant	ETTh1 MSE	ETTh1 MAE	ETTh2 MSE	ETTh2 MAE	Weather MSE	Weather MAE	Electricity MSE	Electricity MAE
Full MARO	**0.396**	**0.423**	**0.316**	**0.373**	0.223	**0.261**	**0.155**	**0.252**
w/o calendar	0.406	0.432	0.333	0.381	**0.223**	0.263	0.161	0.255
w/o channel mixing	0.408	0.433	0.371	0.408	0.224	0.261	0.157	0.256
w/o inter-patch mixer	0.422	0.440	0.332	0.380	0.228	0.266	0.157	0.255
w/o intra-patch mixer	0.407	0.431	0.337	0.384	0.228	0.266	0.168	0.263
Anchor only (no scan)	0.416	0.438	0.337	0.385	0.232	0.267	0.163	0.261
Full-window MLP	0.416	0.436	0.346	0.390	0.225	0.263	0.159	0.259
w/o refinement	0.421	0.442	0.344	0.389	0.225	0.263	0.159	0.256

4.3 Recent-to-Old and Old-to-Recent Traversal

Table 4 compares the two complete traversal conventions on ETTh1, ETTm1, and Weather at the 96- and 336-step horizons. Recent-to-old (R→O) traversal anchors at the most recent patch and scans toward the oldest; old-to-recent (O→R) traversal anchors at the oldest patch and scans toward the most recent. Each processes every patch once. Recent-to-old traversal obtains lower MSE in five of the six settings. The largest gaps occur on ETTm1: MSE falls from 0.3136 to 0.2842 at $H = 96$

Table 4: Test MSE for both traversal directions at two horizons. Lower is better.

	ETTh1		ETTm1		Weather	
Traversal	$H = 96$	$H = 336$	$H = 96$	$H = 336$	$H = 96$	$H = 336$
R→O (MARO)	**0.3554**	0.3959	**0.2842**	**0.3599**	**0.1460**	**0.2442**
O→R	0.3816	**0.3951**	0.3136	0.3846	0.1558	0.2592

and from 0.3846 to 0.3599 at $H = 336$. The exception is ETTh1 at $H = 336$, where old-to-recent traversal is lower by 0.0008.

Computational cost. Figure 5 compares forward FLOPs, inference latency, and peak training memory, and Table 6 in Appendix B.3 gives exact measurements and epoch times. All use $L = 512$, $H = 96$, and one RTX 4060 Laptop GPU. On Electricity, MARO trains $12.15\times$ faster than PatchTST and $12.82\times$ faster than TimeKAN, with about one sixth of PatchTST's parameters and one hundredth of its FLOPs. Its recurrent scan shares weights across channels. On Weather, only DLinear uses fewer FLOPs, while MARO's epoch time, latency, and memory remain below those of TimeKAN and PatchTST. On seven-channel ETTh1, however, the sequential scan under-utilizes the GPU, so MARO trains more slowly than DLinear, iTransformer, and PatchTST despite using fewer FLOPs than PatchTST.

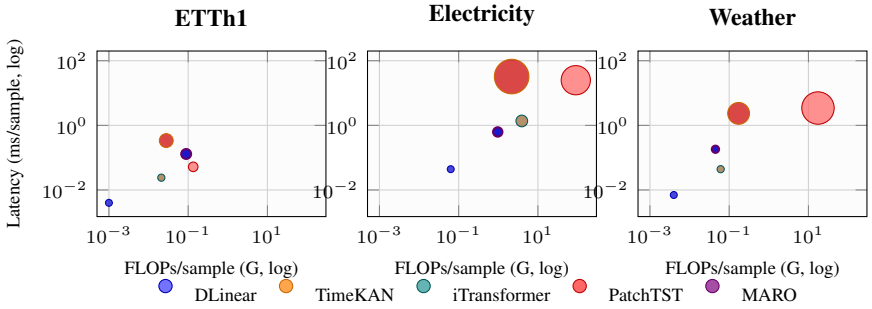


Figure 5: Computational-efficiency comparison under the common protocol detailed in Appendix B.3. Both axes use logarithmic scales. Points closer to the lower-left require fewer FLOPs and have lower inference latency. Bubble size encodes peak training memory, and color identifies the model.

5 Conclusion

We presented MARO, which anchors a forecast at the most recent patch and recurrently integrates older patches. It achieves strong accuracy across long-term and short-term benchmarks, especially on Electricity, Traffic, and shorter PEMS horizons. Ablations identify the recent anchor and progressive integration as the main contributors. Scan direction provides a smaller consistent gain, while checkpoint fusion combines representations across depths.

MARO also offers a favorable accuracy-efficiency trade-off. On Electricity and Weather, it trains faster and uses fewer FLOPs and less memory than PatchTST. It is also faster than TimeKAN in the measured settings. However, MARO uses more parameters than DLinear and TimeKAN, and its sequential computation can be slower than DLinear or iTransformer on small-channel data. Its benefit is forecasting accuracy at moderate computational cost rather than minimum latency.

Limitations & Future Work. The sequential scan can under-utilize accelerators at small batch or channel counts, which motivates parallel or chunked implementations. Learned channel neighbourhoods are fixed across windows and do not consistently outperform fixed ones, so input-adaptive routing merits study. Tests on irregularly sampled series and distribution shifts would assess the generality of recency-ordered computation.

AI USE STATEMENT

Generative AI tools were used to draft parts of the manuscript, revise and polish the writing, and assist with the retrieval and discovery of related work. The authors checked suggested literature against the original sources and manually reviewed all AI-assisted text. Generative AI tools were also used in a supporting role to check implementation consistency, experimental setup, and the completeness of the model and accompanying code. They were not used to make the core research decisions, determine the model design, generate datasets, or independently interpret the experimental results. The authors reviewed and validated all implementation and experimental details and take responsibility for the final content of the paper, including its text, claims, citations, code, and artifacts.

REPRODUCIBILITY STATEMENT

Section 3 specifies the MARO architecture, and Appendix A provides the complete forward pass in pseudocode. Section 4 and Appendices B–D document the experimental protocol, search space, selected configurations, and full results. Additional mechanistic controls and the proof are provided in Appendices E–F. The anonymous code repository contains the training and evaluation code and configurations used for the reported experiments.

REFERENCES

Ibram Abdelmalak, Kiran Madhusudhanan, Jungmin Choi, Christian Klötergens, Vijaya Krishna Yalavarthi, Maximilian Stubbemann, and Lars Schmidt-Thieme. Channel dependence, limited lookback windows, and the simplicity of datasets: How biased is time series forecasting? In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 585–597. Springer, 2026.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, 2019. doi: 10.1145/3292500.3330701.

Musleh Alharthi and Ausif Mahmood. xLSTMTime: Long-term time series forecasting with xLSTM. *AI*, 5(3):1482–1495, 2024.

California Department of Transportation (Caltrans). Performance measurement system (PeMS). `https://pems.dot.ca.gov/`, 2024. Accessed: 2024-05-20.

Si-An Chen, Chun-Liang Li, Nathanael C. Yoder, Sercan O. Arik, and Tomas Pfister. TSMixer: An all-MLP architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023. URL `https://arxiv.org/abs/2303.06053`.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. TSMixer: Lightweight MLP-mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 459–469, 2023. doi: 10.1145/3580305.3599533.

Jingru Fei, Kun Yi, Wei Fan, Qi Zhang, and Zhendong Niu. Amplifier: Bringing attention to neglected low-energy components in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 11645–11653, 2025.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *Advances in Neural Information Processing Systems*, 2025.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E. Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *The 2nd Conference on Language Modeling (COLM)*, 2025. URL `https://openreview.net/forum?id=Itxz7S4Ip3`.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. TimeFilter: Patch-specific spatial-temporal graph filtration for time series forecasting. In *International Conference on Machine Learning*, 2025.

Qihe Huang, Lei Shen, Ruixin Zhang, Jiahuan Cheng, Shouhong Ding, Zhengyang Zhou, and Yang Wang. HDMixer: Hierarchical dependency with extendable patch for multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 12608–12616, 2024.

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

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 *International Conference on Learning Representations*, 2022.

Maurice Kraus, Felix Divo, Devendra Singh Dhami, and Kristian Kersting. xLSTM-Mixer: Multivariate time series forecasting by mixing via scalar memories. In *Advances in Neural Information Processing Systems*, volume 38, pp. 7335–7369, 2025. doi: 10.52202/085713-0228.

Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. CycleNet: Enhancing time series forecasting through modeling periodic patterns. *Advances in Neural Information Processing Systems*, 37:106315–106345, 2024.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. SegRNN: Segment recurrent neural network for long-term time series forecasting. *IEEE Internet of Things Journal*, 13(5):9861–9871, 2026. doi: 10.1109/JIOT.2025.3647705.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. SCINet: Time series modeling and forecasting with sample convolution and interaction. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

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

Max Planck Institute for Biogeochemistry. Jena climate dataset. `https://www.bgc-jena.mpg.de/wetter/`, 2024. Accessed: 2024-05-20.

Md Mahmuddun Nabi Murad, Mehmet Aktukmak, and Yasin Yilmaz. WPMixer: Efficient multi-resolution mixing for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

National Renewable Energy Laboratory. Solar power data for integration studies. `https://www.nrel.gov/grid/solar-power-data.html`, 2024. Accessed: 2024-05-20.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Artur Trindade. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015. DOI: https://doi.org/10.24432/C58C86.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024.

Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Xingjian Wu, Xiangfei Qiu, Hanyin Cheng, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Enhancing time series forecasting through selective representation spaces: A patch perspective. *Advances in Neural Information Processing Systems*, 38:23328–23354, 2025.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

A MARO INFERENCE IN PSEUDOCODE

Algorithm 1 MARO forecast for one input window

Require: $\mathbf{X}, \mathbf{M}^{\mathrm{hist}}, \mathbf{M}^{\mathrm{fut}}; P, S, H, \kappa, R$
Ensure: forecast $\widehat{\mathbf{Y}} \in \mathbb{R}^{C \times H}$

$\widetilde{\mathbf{X}}, \mathcal{S} \leftarrow \mathrm{RevINNorm}(\mathbf{X})$
$\mathbf{p}_{c,1:N} \leftarrow \mathrm{LeftPadPatch}(\widetilde{\mathbf{X}}_c, P, S)$ for all c
$b_{1:N} \leftarrow \mathrm{HistCal}(\mathbf{M}^{\mathrm{hist}}, P, S)$
$T_{c,n} \leftarrow W_P\big(\mathbf{p}_{c,n} + \mathrm{MLP}_{\mathrm{intra}}(\mathbf{p}_{c,n})\big)$ for all c, n
$T_{c,n} \leftarrow T_{c,n} + \mathrm{MLP}_{\mathrm{tok}}(T_{c,n}) + b_n$ for all c, n
if $\kappa > 0$ **then**
  $(\mathcal{N}_c, w_{c,\cdot})_{c=1}^{C} \leftarrow \mathrm{SparseGraph}_{\kappa}()$
  $A_{c,n} \leftarrow \sum_{j \in \mathcal{N}_c \cup \{c\}} w_{c,j} T_{j,n}$ for all c, n
  $U_{c,n} \leftarrow T_{c,n} + g_{\mathrm{ch}}\, \mathrm{MLP}_v(A_{c,n} - T_{c,n})$ for all c, n
else
  $U_{c,n} \leftarrow T_{c,n}$ for all c, n
end if
$(f_{1:H}, \phi) \leftarrow \mathrm{FutureCal}(\mathbf{M}^{\mathrm{fut}})$
$\mathcal{D} \leftarrow \{2^j : 0 \le j \le \lfloor \log_2 N \rfloor\} \cup \{N\}$
for $c = 1$ **to** C **do**
  $a \leftarrow U_{c,N}$; $h \leftarrow a$
  $z \leftarrow a + \mathrm{MLP}_{\mathrm{init}}([a \| \phi])$; $Z_1 \leftarrow z$
  for $s = 1$ **to** $N - 1$ **do**
    $\mathrm{pos}_s \leftarrow \mathrm{Step}(s, N)$
    $\rho_s \leftarrow \mathrm{MLP}_{\mathrm{rel}}([h \| U_{c,N-s} \| U_{c,N-s} \odot h \| \mathrm{pos}_s])$
    $h \leftarrow h + g_r \rho_s$
    $z \leftarrow z + g_r\, \mathrm{MLP}_{\mathrm{upd}}([z \| h \| \rho_s \| a \| \phi \| \mathrm{pos}_s])$
    $Z_{s+1} \leftarrow z$
  end for
  $\mathrm{pos}_\star \leftarrow \mathrm{Step}(\max(1, N-1), N)$
  for $r = 1$ **to** R **do**
    $z \leftarrow z + g_r\, \mathrm{MLP}_{\mathrm{upd}}([z \| h \| \mathbf{0} \| a \| \phi \| \mathrm{pos}_\star])$
  end for
  $Z_N \leftarrow z$; $\bar{z}_c \leftarrow \sum_{k \in \mathcal{D}} \alpha_k Z_k$, where $\alpha = \mathrm{softmax}(\ell^{\mathrm{fuse}})$
  $\widehat{\mathbf{Y}}^{\mathrm{norm}}_{c,:} \leftarrow \mathrm{MLP}_{\mathrm{dec}}(\bar{z}_c) + W_{\mathrm{raw}} \widetilde{\mathbf{X}}_c + \big(v_c^\top f_t + b_c\big)_{t=1}^{H}$
end for
$\widehat{\mathbf{Y}} \leftarrow \mathrm{RevINDenorm}(\widehat{\mathbf{Y}}^{\mathrm{norm}}, \mathcal{S})$
return $\widehat{\mathbf{Y}}$

Algorithm 1 gives the forward pass for one window, with channel operations written in parallel. $\mathrm{LeftPadPatch}$ zero-pads the oldest side so the last patch ends at the forecast boundary. $\mathrm{HistCal}$ appends the relative-position cycle features described in Appendix B.1 and encodes each patch's first, middle, and last calendar descriptors as b_n. $\mathrm{FutureCal}$ appends the same cycle features for the horizon, returns one token f_t per future step, and encodes the first, middle, and last future tokens into the shared context ϕ. The same b_n and ϕ are used for every channel.

$\mathrm{SparseGraph}_{\kappa}$ returns the κ highest-scoring other channels for each target and normalized weights over those channels and the target itself; the learned self-score participates in normalization (Appendix E.1). When $\kappa = 0$, channel mixing is skipped. The final refined state replaces the deepest checkpoint before fusion. When $N = 1$, optional refinement uses position index one with denominator one in Step, matching the implementation.

B EXPERIMENTAL SETUP

B.1 BENCHMARKS AND EVALUATION PROTOCOL

Table 5 summarizes the twelve benchmarks and their chronological splits. We follow the Informer protocol (Zhou et al., 2021) for the long-term benchmarks and the CycleNet protocol (Lin et al.,

2024) for PEMS. Standardization statistics are fitted on the training split, and sliding windows are constructed with stride one. MARO applies RevIN within each input window.

Table 5: Benchmarks used in this work. C is the number of variables and T is the number of time steps. The final column gives the chronological train-validation-test split under the published benchmark protocols.

Dataset	Domain	C	T	Sampling	Split (train/val/test)
ETTh1 (Zhou et al., 2021)	Transformer load	7	17,420	1 hour	8,640 / 2,880 / 2,880
ETTh2 (Zhou et al., 2021)	Transformer load	7	17,420	1 hour	8,640 / 2,880 / 2,880
ETTm1 (Zhou et al., 2021)	Transformer load	7	69,680	15 min	34,560 / 11,520 / 11,520
ETTm2 (Zhou et al., 2021)	Transformer load	7	69,680	15 min	34,560 / 11,520 / 11,520
Weather (Max Planck Institute for Biogeochemistry, 2024)	Meteorology	21	52,696	10 min	36,887 / 5,270 / 10,539
Electricity (Trindade, 2015)	Power demand	321	26,304	1 hour	18,412 / 2,632 / 5,260
Traffic (California Department of Transportation (Caltrans), 2024)	Road occupancy	862	17,544	1 hour	12,280 / 1,756 / 3,508
Solar (National Renewable Energy Laboratory, 2024)	Photovoltaic output	137	52,560	10 min	36,792 / 5,256 / 10,512
PEMS03 (California Department of Transportation (Caltrans), 2024)	Traffic flow	358	26,208	5 min	15,724 / 5,241 / 5,241
PEMS04 (California Department of Transportation (Caltrans), 2024)	Traffic flow	307	16,992	5 min	10,195 / 3,398 / 3,398
PEMS07 (California Department of Transportation (Caltrans), 2024)	Traffic flow	883	28,224	5 min	16,934 / 5,644 / 5,644
PEMS08 (California Department of Transportation (Caltrans), 2024)	Traffic flow	170	17,856	5 min	10,713 / 3,571 / 3,571

As an implementation detail, MARO augments timestamp-derived calendar features with relative-position cycle features. For an offset r from the forecast origin, we append $\sin(2\pi r/W)$ and $\cos(2\pi r/W)$, using $r = -L+1, \dots, 0$ for the historical window and $r = 1, \dots, H$ for the forecast horizon. The dataset-specific cycle length W is 24 for ETTh1/ETTh2, 96 for ETTm1/ETTm2, 144 for Weather and Solar, and 168 for Electricity (ECL) and Traffic.

B.2 Error Measures

Write $Y_{i,h,c}$ for a target and $\widehat{Y}_{i,h,c}$ for the corresponding forecast. Index i runs over the n test windows. Index h runs over the horizon and c runs over the channels. The reported metrics are

$$\mathrm{MSE} = \frac{1}{nHC}\sum_{i=1}^{n}\sum_{h=1}^{H}\sum_{c=1}^{C}\left(\widehat{Y}_{i,h,c} - Y_{i,h,c}\right)^2, \qquad \mathrm{MAE} = \frac{1}{nHC}\sum_{i=1}^{n}\sum_{h=1}^{H}\sum_{c=1}^{C}\left|\widehat{Y}_{i,h,c} - Y_{i,h,c}\right|. \tag{11}$$

MSE is used for training and model selection. Both metrics are averaged over all channels and horizon steps. Lower values are better.

B.3 Efficiency Measurements

Table 6 reports the exact measurements visualized in Figure 5, together with parameter counts and first-epoch training time.

C Hyperparameter Selection

C.1 Search Space and Budget

For each dataset–horizon setting, we run one 30-trial Optuna study (Akiba et al., 2019). The long-term look-back length $L \in \{96, 336, 512\}$ is optimized jointly with the remaining hyperparameters within those same 30 trials; PEMS fixes $L = 96$. Each selected configuration is evaluated with random seeds 3000, 3002, and 3004. Table 7 gives the search coordinates and fixed settings. Weight decay is selected from the listed values for the long-term benchmarks and fixed at zero for PEMS.

C.2 Selected Configurations

Tables 8 and 9 report the best backbone and optimization configurations for the long-term and PEMS benchmarks, respectively. Table 10 gives the remaining long-term gate settings.

D Full Results

Tables 11 and 12 report the full long-term forecasting results.

Table 6: Measured cost of MARO and four representative baselines under a common protocol with $L = 512$ and $H = 96$ on one RTX 4060 Laptop GPU. Params counts trainable parameters, FLOPs is the forward cost of one sample and FLOPs$/C$ the same quantity per input channel, Epoch is the wall-clock time of the first training epoch, Latency is the evaluation-mode forward time per sample, and Memory is the peak allocation during training. ETTh1 uses batch size 32, Electricity uses batch size 4, and Weather uses batch size 16. Lower is better throughout.

Dataset	Model	Params (M)	FLOPs (G)	FLOPs/C (G/ch.)	Epoch (s)	Latency (ms/sample)	Memory (MB)
ETTh1 ($C = 7$, batch size 32)							
	DLinear	0.10	0.001	0.0001	0.62	0.004	21
	TimeKAN	0.05	0.028	0.0040	7.54	0.338	423
	iTransformer	0.95	0.021	0.0030	1.64	0.024	47
	PatchTST	0.94	0.134	0.0191	1.91	0.052	162
	MARO (Ours)	0.95	0.089	0.0128	4.87	0.130	167
Electricity ($C = 321$, batch size 4)							
	DLinear	0.10	0.063	0.0002	12.10	0.044	37
	TimeKAN	0.06	2.170	0.0068	1395.76	32.348	4180
	iTransformer	5.05	3.918	0.0122	81.12	1.368	274
	PatchTST	7.40	90.365	0.2815	1323.39	24.970	2729
	MARO (Ours)	1.15	0.968	0.0030	108.88	0.625	146
Weather ($C = 21$, batch size 16)							
	DLinear	0.10	0.004	0.0002	5.38	0.007	23
	TimeKAN	0.06	0.173	0.0082	200.28	2.348	1408
	iTransformer	1.21	0.061	0.0029	19.04	0.044	55
	PatchTST	9.46	17.418	0.8294	486.50	3.438	3600
	MARO (Ours)	0.42	0.045	0.0022	44.11	0.182	57

Table 7: Unified Optuna search space for MARO.

Quantity	Symbol	Range or value
Learning rate	η	log-uniform, $[1 \times 10^{-5}, 5 \times 10^{-3}]$
Weight decay (long-term)	λ	$\{0, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$
Look-back length	L	$\{96, 336, 512\}$
Token width	d	$\{32, 64, 128, 256, 512\}$
Inter-patch hidden width	d_{ff}	$\{64, 128, 256, 512, 1024\}$
Latent MLP layers	L_e	integer in $[1, 4]$
Intra-patch MLP layers	L_d	integer in $[1, 4]$
Dropout		uniform, $[0, 0.40]$
Patch length	P	$\{8, 12, 24, 32, 48, 64, 96, 112, 128, 144, 192, 256\}$, capped at L
Stride ratio	r	$\{0.30, 0.50, 0.75\}$
Stride	S	$\max(1, \text{round}(rP))$, derived
Channel neighbourhood	κ	$\{0, 2, 5, 6, 16, 32, 64, 100, 160, 320\}$, capped at $C - 1$
Channel embedding width	d_g	$\{4, 8, 16, 32, 64, 128\}$, sampled only when $\kappa > 0$
Initial channel gate	g_{ch}	log-uniform, $[10^{-4}, 5 \times 10^{-2}]$, sampled only when $\kappa > 0$
Initial self score	β	$\{0, 0.5, 1, 2\}$, sampled only when $\kappa > 0$
Initial recurrent gate	g_r	$\{0.01, 0.025, 0.05, 0.10, 0.20\}$
Post-scan refinement steps	R	$\{0, 1, 2, 3\}$
Channel gate bound	$\bar{g}_{\text{ch}}$	fixed at 0.20
Recurrent gate bound	$\bar{g}_r$	fixed at 0.30
Weight decay (PEMS)	λ	fixed at 0
Normalization		fixed, RevIN with a learned affine map
Activation		fixed, GELU

Table 8: Best MARO backbone and optimization configurations for the long-term benchmarks. Here λ denotes weight decay.

Dataset	H	L	P	S	d	d_{ff}	L_e	L_d	R	Dropout	η	λ	B
ETTh1	96	512	64	32	32	1024	1	1	1	0.014	2.645×10^{-4}	1.0×10^{-5}	256
ETTh1	192	512	64	32	32	1024	1	4	0	0.082	3.677×10^{-4}	0	256
ETTh1	336	512	144	108	64	64	1	1	2	0.129	6.353×10^{-4}	1.0×10^{-7}	256
ETTh1	720	336	112	84	128	128	1	1	1	0.267	1.812×10^{-3}	1.0×10^{-3}	256
ETTh2	96	512	32	16	32	512	4	1	3	0.320	4.370×10^{-4}	0	256
ETTh2	192	512	128	64	32	1024	1	1	3	0.166	6.428×10^{-4}	1.0×10^{-6}	256
ETTh2	336	512	64	19	32	1024	1	1	0	0.009	1.510×10^{-4}	1.0×10^{-3}	256
ETTh2	720	512	128	64	32	512	2	2	3	0.098	2.890×10^{-4}	0	256
ETTm1	96	512	96	72	32	1024	1	1	0	0.385	1.110×10^{-3}	0	256
ETTm1	192	512	192	144	512	64	1	1	3	0.125	4.038×10^{-4}	1.0×10^{-4}	256
ETTm1	336	512	96	48	32	512	2	2	1	0.077	2.498×10^{-4}	0	256
ETTm1	720	512	256	128	32	1024	1	1	1	0.081	1.402×10^{-4}	1.0×10^{-5}	256
ETTm2	96	336	24	7	64	512	1	1	0	0.359	2.150×10^{-4}	0	256
ETTm2	192	512	12	4	32	256	4	2	2	0.341	1.345×10^{-4}	0	256
ETTm2	336	512	48	24	32	64	1	1	3	0.305	6.552×10^{-4}	1.0×10^{-3}	256
ETTm2	720	512	24	12	32	256	1	1	2	0.110	2.617×10^{-4}	1.0×10^{-7}	256
Weather	96	512	192	96	32	128	3	3	0	0.192	4.722×10^{-4}	0	64
Weather	192	512	144	72	64	1024	4	2	2	0.003	5.615×10^{-5}	0	64
Weather	336	512	144	72	32	512	3	2	2	0.390	7.942×10^{-5}	0	64
Weather	720	512	12	9	64	1024	3	4	2	0.153	2.385×10^{-5}	0	64
Electricity	96	512	256	192	128	128	3	1	0	0.152	6.634×10^{-4}	0	16
Electricity	192	512	96	72	256	128	2	1	2	0.297	3.393×10^{-3}	0	16
Electricity	336	512	256	128	256	1024	3	2	1	0.265	9.763×10^{-4}	0	16
Electricity	720	512	96	72	32	1024	4	2	2	0.258	1.650×10^{-3}	0	16
Traffic	96	512	192	144	256	1024	2	3	2	0.004	8.215×10^{-4}	0	16
Traffic	192	512	256	77	128	512	1	4	3	0.336	2.212×10^{-3}	0	16
Traffic	336	512	144	108	256	1024	4	3	0	0.068	5.055×10^{-4}	0	16
Traffic	720	512	192	96	256	1024	3	1	3	0.066	3.667×10^{-4}	0	16
Solar	96	512	96	48	256	256	1	3	3	0.309	1.387×10^{-3}	0	64
Solar	192	512	96	48	64	256	3	2	2	0.067	4.518×10^{-4}	0	16
Solar	336	512	12	6	512	256	1	3	0	0.243	7.979×10^{-5}	0	16
Solar	720	512	12	4	32	256	1	4	3	0.233	6.316×10^{-4}	0	16

Table 9: Best MARO configurations for the PEMS benchmarks.

Dataset	H	P	S	d	d_{ff}	L_e	L_d	R	Dropout	η	κ	$\kappa/(C-1)$	d_g	β	g_{ch} init	g_r init	B
PEMS03	12	96	29	64	256	3	4	1	0.133	2.523×10^{-3}	320	0.896	64	2.00	8.30×10^{-3}	0.100	16
PEMS03	24	12	6	64	512	1	3	0	0.109	4.786×10^{-3}	320	0.896	4	0.00	5.01×10^{-3}	0.100	16
PEMS03	48	12	9	256	1024	1	3	0	0.109	8.417×10^{-4}	6	0.017	128	2.00	3.02×10^{-3}	0.200	16
PEMS03	96	32	10	128	1024	1	3	0	0.185	2.426×10^{-3}	100	0.280	64	2.00	4.75×10^{-3}	0.010	16
PEMS04	12	12	6	512	1024	1	2	1	0.197	1.794×10^{-3}	32	0.105	4	2.00	1.91×10^{-2}	0.200	16
PEMS04	24	48	36	128	1024	2	4	2	0.034	2.284×10^{-3}	100	0.327	64	1.00	2.64×10^{-2}	0.025	16
PEMS04	48	24	7	64	1024	2	4	3	0.271	2.598×10^{-3}	5	0.016	64	0.00	3.19×10^{-4}	0.200	16
PEMS04	96	48	14	512	256	3	3	1	0.133	1.441×10^{-3}	32	0.105	8	0.00	4.74×10^{-2}	0.050	16
PEMS07	12	12	6	128	512	1	1	3	0.022	3.094×10^{-3}	16	0.018	32	0.50	2.28×10^{-2}	0.200	8
PEMS07	24	48	24	512	128	2	2	1	0.148	1.336×10^{-3}	100	0.113	128	0.50	5.12×10^{-4}	0.200	8
PEMS07	48	48	24	512	512	4	2	1	0.295	3.891×10^{-4}	16	0.018	16	0.50	1.47×10^{-2}	0.050	8
PEMS07	96	24	7	128	1024	1	1	2	0.249	1.082×10^{-3}	320	0.363	32	0.50	3.05×10^{-4}	0.025	8
PEMS08	12	48	14	256	1024	2	2	1	0.285	8.295×10^{-4}	32	0.189	128	0.50	2.58×10^{-2}	0.025	16
PEMS08	24	12	4	256	1024	1	3	0	0.031	2.083×10^{-3}	5	0.030	16	2.00	4.77×10^{-4}	0.200	16
PEMS08	48	8	4	32	512	2	2	2	0.068	2.332×10^{-3}	64	0.379	128	2.00	7.33×10^{-4}	0.100	16
PEMS08	96	12	9	512	1024	2	2	2	0.167	1.342×10^{-3}	2	0.012	32	0.00	4.69×10^{-2}	0.200	16

Table 10: Best MARO channel-mixer and gate configurations for the long-term benchmarks.

Dataset	H	κ	$\kappa/(C-1)$	d_g	β	g_{ch} init	g_r init
ETTh1	96	2	0.333	16	1.00	8.93×10^{-4}	0.200
ETTh1	192	6	1.000	8	0.00	1.38×10^{-2}	0.010
ETTh1	336	6	1.000	32	1.00	4.72×10^{-2}	0.100
ETTh1	720	2	0.333	16	1.00	3.66×10^{-4}	0.010
ETTh2	96	5	0.833	128	2.00	3.99×10^{-3}	0.050
ETTh2	192	5	0.833	4	0.50	6.16×10^{-3}	0.010
ETTh2	336	6	1.000	64	1.00	2.52×10^{-3}	0.100
ETTh2	720	5	0.833	8	2.00	1.16×10^{-3}	0.010
ETTm1	96	5	0.833	128	1.00	2.30×10^{-2}	0.010
ETTm1	192	5	0.833	64	1.00	2.24×10^{-3}	0.050
ETTm1	336	5	0.833	32	1.00	5.64×10^{-4}	0.100
ETTm1	720	2	0.333	4	2.00	2.70×10^{-3}	0.200
ETTm2	96	5	0.833	4	0.00	7.37×10^{-3}	0.050
ETTm2	192	2	0.333	64	2.00	8.65×10^{-4}	0.010
ETTm2	336	5	0.833	8	1.00	2.23×10^{-4}	0.025
ETTm2	720	2	0.333	16	2.00	5.48×10^{-3}	0.200
Weather	96	5	0.250	8	0.00	1.22×10^{-2}	0.100
Weather	192	16	0.800	8	2.00	5.72×10^{-4}	0.050
Weather	336	6	0.300	64	1.00	2.29×10^{-3}	0.050
Weather	720	5	0.250	4	0.50	5.30×10^{-4}	0.050
Electricity	96	64	0.200	128	0.00	1.26×10^{-2}	0.100
Electricity	192	2	0.006	4	2.00	9.37×10^{-4}	0.025
Electricity	336	100	0.312	32	1.00	3.08×10^{-3}	0.100
Electricity	720	100	0.312	32	1.00	4.26×10^{-4}	0.010
Traffic	96	64	0.074	32	0.00	1.06×10^{-4}	0.025
Traffic	192	5	0.006	16	1.00	4.41×10^{-3}	0.100
Traffic	336	32	0.037	64	0.50	1.87×10^{-2}	0.050
Traffic	720	6	0.007	8	2.00	8.99×10^{-4}	0.200
Solar	96	16	0.118	128	1.00	1.60×10^{-4}	0.050
Solar	192	6	0.044	4	0.50	3.85×10^{-4}	0.050
Solar	336	32	0.235	128	2.00	4.61×10^{-2}	0.100
Solar	720	6	0.044	16	0.00	1.85×10^{-4}	0.010

Table 11: Full long-term forecasting results. The remaining baselines are shown in Table 12.

		MARO (Ours)		SRSNet		TimeKAN		Amplifier	
Dataset	Horizon	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.354** ±0.0005	**0.390** ±0.0004	0.366	0.394	0.370	0.396	0.373	0.399
	192	**0.391** ±0.0008	**0.411** ±0.0010	0.400	0.415	0.403	0.417	0.414	0.420
	336	**0.398** ±0.0019	**0.429** ±0.0027	0.424	0.430	0.420	0.432	0.442	0.446
	720	0.440 ±0.0030	0.460 ±0.0024	**0.426**	**0.455**	0.442	0.465	0.455	0.467
Avg		**0.396**	**0.423**	0.404	**0.423**	0.408	0.427	0.421	0.433
ETTh2	96	**0.254** ±0.0018	**0.324** ±0.0011	0.271	0.338	0.280	0.343	0.287	0.349
	192	**0.312** ±0.0043	**0.368** ±0.0028	0.324	0.372	0.329	0.382	0.348	0.393
	336	**0.314** ±0.0004	**0.376** ±0.0002	0.349	0.394	0.370	0.412	0.383	0.423
	720	**0.383** ±0.0038	**0.425** ±0.0032	0.391	0.434	0.420	0.450	0.407	0.444
Avg		**0.316**	**0.373**	0.333	0.384	0.349	0.396	0.356	0.402
ETTm1	96	**0.287** ±0.0033	0.346 ±0.0019	0.288	**0.341**	0.290	0.348	0.292	0.346
	192	0.330 ±0.0007	0.368 ±0.0006	**0.326**	**0.364**	0.332	0.368	0.327	0.365
	336	0.360 ±0.0038	0.385 ±0.0012	0.365	0.386	**0.354**	0.386	0.365	0.386
	720	0.421 ±0.0009	**0.416** ±0.0006	0.426	0.420	**0.401**	0.417	0.427	0.419
Avg		0.349	0.379	0.351	**0.377**	**0.344**	0.379	0.352	0.379
ETTm2	96	**0.163** ±0.0004	**0.251** ±0.0004	0.164	0.254	0.164	0.254	0.164	0.254
	192	**0.218** ±0.0007	**0.291** ±0.0002	0.220	0.296	0.238	0.300	0.226	0.300
	336	0.275 ±0.0019	0.328 ±0.0024	**0.271**	**0.327**	0.278	0.331	0.276	0.331
	720	0.355 ±0.0020	0.381 ±0.0011	**0.353**	**0.380**	0.359	0.387	0.358	0.388
Avg		0.253	**0.313**	**0.252**	0.314	0.259	0.318	0.256	0.318
Weather	96	**0.145** ±0.0008	**0.194** ±0.0004	0.147	0.198	0.151	0.205	0.147	0.199
	192	0.191 ±0.0007	0.240 ±0.0009	**0.190**	0.242	0.195	0.244	0.194	0.245
	336	0.242 ±0.0009	**0.278** ±0.0008	**0.241**	0.282	0.242	0.287	0.243	0.282
	720	0.315 ±0.0013	0.331 ±0.0006	0.325	0.340	0.317	0.340	**0.310**	**0.329**
Avg		**0.223**	**0.261**	0.225	0.265	0.226	0.269	**0.223**	0.263
Electricity	96	**0.126** ±0.0004	**0.223** ±0.0004	0.131	0.226	0.135	0.231	0.132	0.227
	192	**0.145** ±0.0006	**0.241** ±0.0003	0.147	**0.241**	0.149	0.243	0.149	0.243
	336	**0.162** ±0.0027	0.262 ±0.0022	0.163	**0.258**	0.165	0.260	0.167	0.261
	720	**0.186** ±0.0054	**0.282** ±0.0033	0.201	0.291	0.206	0.297	0.203	0.292
Avg		**0.155**	**0.252**	0.160	0.254	0.163	0.257	0.162	0.255
Traffic	96	**0.340** ±0.0031	**0.252** ±0.0013	0.361	0.254	0.388	0.269	0.391	0.277
	192	**0.351** ±0.0004	**0.254** ±0.0006	0.380	0.263	0.411	0.286	0.405	0.283
	336	**0.380** ±0.0022	**0.269** ±0.0027	0.392	0.270	0.425	0.284	0.416	0.290
	720	**0.418** ±0.0026	**0.285** ±0.0018	0.434	0.293	0.455	0.302	0.436	0.312
Avg		**0.372**	**0.265**	0.391	0.270	0.419	0.285	0.412	0.290
Solar	96	0.169 ±0.0107	0.223 ±0.0137	**0.167**	**0.222**	0.187	0.255	0.175	0.237
	192	0.190 ±0.0056	0.251 ±0.0080	**0.182**	**0.237**	0.194	0.265	0.198	0.259
	336	**0.187** ±0.0035	**0.245** ±0.0013	0.188	**0.245**	0.203	0.264	0.213	0.259
	720	0.198 ±0.0019	0.261 ±0.0061	**0.195**	**0.251**	0.209	0.269	0.222	0.369
Avg		0.186	0.245	**0.183**	**0.238**	0.198	0.263	0.202	0.281
# Best / Total		**20/32**	**20/32**	9/32	11/32	2/32	0/32	1/32	1/32

Table 12: Full long-term forecasting results continued from Table 11.

		MARO (Ours)		iTransformer		TimeMixer		PatchTST	
Dataset	Horizon	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.354** ±0.0005	**0.390** ±0.0004	0.386	0.405	0.372	0.401	0.377	0.397
	192	**0.391** ±0.0008	**0.411** ±0.0010	0.430	0.435	0.413	0.430	0.409	0.425
	336	**0.398** ±0.0019	**0.429** ±0.0027	0.450	0.452	0.438	0.450	0.431	0.444
	720	0.440 ±0.0030	0.460 ±0.0024	0.495	0.487	0.483	0.483	0.457	0.477
Avg		**0.396**	**0.423**	0.440	0.444	0.426	0.441	0.418	0.435
ETTh2	96	**0.254** ±0.0018	**0.324** ±0.0011	0.292	0.347	0.270	0.342	0.274	0.337
	192	**0.312** ±0.0043	**0.368** ±0.0028	0.348	0.384	0.349	0.387	0.348	0.384
	336	**0.314** ±0.0004	**0.376** ±0.0002	0.372	0.407	0.367	0.410	0.377	0.416
	720	**0.383** ±0.0038	**0.425** ±0.0032	0.424	0.444	0.401	0.436	0.406	0.441
Avg		**0.316**	**0.373**	0.359	0.395	0.346	0.393	0.351	0.394
ETTm1	96	**0.287** ±0.0033	0.346 ±0.0019	**0.287**	0.342	0.293	0.345	0.289	0.343
	192	0.330 ±0.0007	0.368 ±0.0006	0.331	0.371	0.335	0.372	0.329	0.368
	336	0.360 ±0.0038	0.385 ±0.0012	0.358	**0.384**	0.368	0.386	0.362	0.390
	720	0.421 ±0.0009	**0.416** ±0.0006	0.412	**0.416**	0.426	0.417	0.416	0.423
Avg		0.349	0.379	0.347	0.378	0.355	0.380	0.349	0.381
ETTm2	96	**0.163** ±0.0004	**0.251** ±0.0004	0.168	0.262	0.165	0.256	0.165	0.255
	192	**0.218** ±0.0007	**0.291** ±0.0002	0.224	0.295	0.225	0.298	0.221	0.293
	336	0.275 ±0.0019	0.328 ±0.0024	0.274	0.330	0.277	0.332	0.276	**0.327**
	720	0.355 ±0.0020	0.381 ±0.0011	0.367	0.385	0.360	0.387	0.362	0.381
Avg		0.253	**0.313**	0.258	0.318	0.256	0.318	0.256	0.314
Weather	96	**0.145** ±0.0008	**0.194** ±0.0004	0.150	0.207	0.151	0.198	0.150	0.200
	192	0.191 ±0.0007	0.240 ±0.0009	0.200	0.248	0.191	0.242	0.191	**0.239**
	336	0.242 ±0.0009	**0.278** ±0.0008	0.252	0.287	0.244	0.280	0.242	0.279
	720	0.315 ±0.0013	0.331 ±0.0006	0.320	0.336	0.316	0.331	0.312	0.330
Avg		**0.223**	**0.261**	0.230	0.269	0.225	0.262	**0.223**	0.262
Electricity	96	**0.126** ±0.0004	**0.223** ±0.0004	0.134	0.230	0.153	0.256	0.143	0.247
	192	**0.145** ±0.0006	**0.241** ±0.0003	0.154	0.250	0.168	0.269	0.158	0.260
	336	**0.162** ±0.0027	0.262 ±0.0022	0.169	0.265	0.189	0.281	0.168	0.267
	720	**0.186** ±0.0054	**0.282** ±0.0033	0.194	0.288	0.228	0.320	0.214	0.307
Avg		**0.155**	**0.252**	0.162	0.258	0.184	0.281	0.170	0.270
Traffic	96	**0.340** ±0.0031	**0.252** ±0.0013	0.363	0.265	0.369	0.256	0.370	0.262
	192	**0.351** ±0.0004	**0.254** ±0.0006	0.384	0.273	0.400	0.271	0.386	0.269
	336	**0.380** ±0.0022	**0.269** ±0.0027	0.396	0.277	0.407	0.272	0.396	0.275
	720	**0.418** ±0.0026	**0.285** ±0.0018	0.445	0.308	0.462	0.316	0.435	0.295
Avg		**0.372**	**0.265**	0.397	0.280	0.409	0.278	0.396	0.275
Solar	96	0.169 ±0.0107	0.223 ±0.0137	0.174	0.229	0.180	0.233	0.170	0.234
	192	0.190 ±0.0056	0.251 ±0.0080	0.205	0.270	0.201	0.259	0.204	0.302
	336	**0.187** ±0.0035	**0.245** ±0.0013	0.216	0.282	0.214	0.272	0.212	0.293
	720	0.198 ±0.0019	0.261 ±0.0061	0.211	0.260	0.218	0.278	0.215	0.307
Avg		0.186	0.245	0.201	0.260	0.203	0.260	0.200	0.284
# Best / Total		**20/32**	**20/32**	1/32	2/32	0/32	0/32	0/32	2/32

D.1 Scan-Direction Update Controls

Table 13 reports the full set of RNN and causal attention scan-direction results behind Figure 2. Values are truncated to three decimals, so close comparisons such as the Weather RNN MSE appear tied at the displayed precision.

Table 13: Scan-direction update controls on all five datasets. Each entry is test MSE or MAE truncated toward zero to three decimal places. Lower is better.

	RNN update				Causal attention			
	Recent→old		Old→recent		Recent→old		Old→recent	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.365	0.400	0.372	0.401	0.364	0.398	0.379	0.410
ETTh2	0.279	0.341	0.268	0.337	0.260	0.327	0.260	0.331
ETTm1	0.298	0.356	0.315	0.364	0.293	0.346	0.311	0.355
ETTm2	0.171	0.255	0.165	0.255	0.165	0.253	0.167	0.256
Weather	0.151	0.202	0.151	0.205	0.147	0.197	0.157	0.208

E Method Details and Mechanistic Controls

E.1 Sparse Channel-Mixing Formulation

For completeness, this section gives the channel-mixing equations summarized in Section 3.3. Each channel has a learned embedding $e_c \in \mathbb{R}^{d_g}$. Two MLPs map it to a query $q_c = \mathrm{MLP}_q(e_c)$ and a key $k_c = \mathrm{MLP}_k(e_c)$. The score from channel j to target channel c is

$$G_{c,j} = \frac{q_c^\top k_j}{\sqrt{d_g}} \quad (c \neq j). \tag{12}$$

The κ highest-scoring channels form the neighbourhood $\mathcal{N}_c$. A softmax over their scores and a learned self-score β gives normalized weights $w_{c,j}$. At patch n, the corresponding mixture is

$$A_{c,n} = w_{c,c}\, T_{c,n} + \sum_{j \in \mathcal{N}_c} w_{c,j}\, T_{j,n}. \tag{13}$$

The original token receives a gated residual correction,

$$U_{c,n} = T_{c,n} + g_{\mathrm{ch}} \cdot \mathrm{MLP}_v\big(A_{c,n} - T_{c,n}\big), \tag{14}$$

where $g_{\mathrm{ch}} = \bar{g}_{\mathrm{ch}}\sigma(\ell_{\mathrm{ch}})$ is a bounded learned gate initialized near zero. The normalized weights sum to one, which is why a calendar token shared by all channels cancels from $A_{c,n} - T_{c,n}$.

E.2 Channel Topology

Table 14: Channel-topology controls. Entries are test MSE over three seeds and bold marks the lowest mean in each row.

Cell	Learned	Random fixed	Correlation fixed
Electricity H=96	.12724 ± .00025	.12679 ± .00005	**.12579 ± .00073**
Electricity H=336	**.16202 ± .00162**	.16514 ± .00285	.16738 ± .00266
Solar H=96	.16597 ± .00261	.16466 ± .00560	**.16386 ± .00200**
Solar H=336	**.18707 ± .00277**	.19922 ± .00190	.18878 ± .00098
Traffic H=96	.34486 ± .00584	**.33564 ± .00056**	.33966 ± .00018
Traffic H=336	.38336 ± .00403	**.38083 ± .00140**	.38601 ± .00030

Table 14 compares learned and fixed channel neighbourhoods under the same MARO configuration. Each topology gives the lowest mean MSE in two of the six settings. The benefit of learning the channel graph therefore depends on the dataset rather than holding universally.

Table 15: Gate controls. Entries are three-seed mean MSE changes from the corresponding baseline and negative is better.

Cell	Frozen Δ	LayerScale Δ	Wide Δ	Separate ctx. gate Δ
ETTh1 H=96	$+.000010$	$-.000031$	$-.000002$	$+.000015$
ETTh1 H=336	$+.000009$	$+.000040$	$-.000015$	$+.000052$
ETTh2 H=96	$-.000388$	$+.000350$	$+.000494$	$+.000546$
ETTh2 H=336	$-.000015$	$+.000055$	$-.000011$	$-.000006$
Weather H=96	$+.000863$	$+.000936$	$+.001294$	$+.000671$
Weather H=336	$-.000307$	$-.000263$	$-.000241$	$-.000052$

E.3 Gate Parameterization and Context Gating

The gate alternatives produce only small changes from the bounded-gate baseline. A separate context gate also gives no consistent improvement. MARO therefore retains one shared recurrent gate.

E.4 Checkpoint-Depth Fusion and the Recurrent Gate

We inspect the recurrent gate and checkpoint-fusion weights from the seed-3000 checkpoints for all 32 long-term settings. The recurrent gate remains below its fixed bound, while checkpoint weights are broadly distributed across depths (Table 16).

Table 16: Gate and checkpoint-fusion values from the 32 selected long-term dataset–horizon checkpoints at seed 3000. Here $\bar{g}_r = 0.30$, α_1 and α_N are the shallowest and deepest checkpoint weights, and normalized entropy is $-\sum_k \alpha_k \log \alpha_k / \log |\mathcal{D}|$. The number of checkpoints $|\mathcal{D}|$ varies by configuration.

Quantity	Mean	Median	Min	Max
g_r	0.0701	0.0503	0.0091	0.2008
$g_r/\bar{g}_r$	0.234	0.168	0.030	0.670
α_1	0.230	0.240	0.124	0.340
α_N	0.237	0.246	0.127	0.424
Normalized fusion entropy	0.997	0.9999	0.948	1.000

Because these parameters remain close to their initial values in many settings, Appendix E.5 additionally evaluates the trained model with deepest-state-only inference.

E.5 Deepest-State-Only Inference Diagnostic

At inference, we replace the fused representation with the deepest checkpoint Z_N, leaving the remaining trained model unchanged. We evaluate the same 32 seed-3000 checkpoints.

Table 17: Deepest-state-only inference diagnostic over the 32 long-term dataset–horizon settings. With $d = \mathrm{MSE}_{Z_N} - \mathrm{MSE}_{\mathrm{fused}}$, worsened means $d > 10^{-6}$, improved means $d < -10^{-6}$, and unchanged covers the remaining cells. Mean ΔMSE averages d over all 32 cells.

Control	Changed	Worsened	Improved	Unchanged	Mean ΔMSE
Deepest state only	30/32	26	4	2	$+0.0699$

Tables 17 and 18 summarize the results. Test MSE changes in 30 of 32 cells: 26 worsen, four improve, and two remain unchanged within 10^{-6}. The mean MSE increase is 0.0699. Solar accounts for much of this effect, with a mean increase of 0.4408 across its four horizons.

This single-seed inference intervention changes the representation presented to a decoder trained with checkpoint fusion, so it should be interpreted as a diagnostic rather than as the gain from training with fusion.

Table 18: Deepest-state-only test-MSE change by dataset (four horizons each, seed 3000). Counts use the same 10^{-6} threshold as the aggregate table.

Dataset	Worsened	Improved	Unchanged	Mean ΔMSE	Median ΔMSE	Max ΔMSE
ETTh1	2/4	1/4	1/4	+0.0033	+0.0000	+0.0132
ETTh2	2/4	2/4	0/4	+0.0024	−0.0002	+0.0111
ETTm1	3/4	1/4	0/4	+0.0081	+0.0077	+0.0172
ETTm2	4/4	0/4	0/4	+0.0545	+0.0486	+0.1209
Electricity	3/4	0/4	1/4	+0.0103	+0.0073	+0.0266
Traffic	4/4	0/4	0/4	+0.0119	+0.0049	+0.0376
Weather	4/4	0/4	0/4	+0.0278	+0.0299	+0.0441
Solar	4/4	0/4	0/4	+0.4408	+0.4147	+0.8397

F Proof and Increment Analysis

F.1 Proof of Proposition 1

Fix one channel and the realized pre-gate scan vectors u_s and post-scan vectors v_r defined in Proposition 1. Unrolling the additive latent update of equation 5 gives, for each unrefined scan state at depth k,

$$z^{(k-1)} = z^{(0)} + g_r \sum_{s=1}^{k-1} u_s, \tag{15}$$

because the same g_r multiplies each pre-gate update vector. Only the deepest checkpoint is replaced after R refinement steps. By equation 9, that refined state is

$$Z_N^{\mathrm{ref}} = z^{(0)} + g_r \sum_{s=1}^{N-1} u_s + g_r \sum_{r=1}^{R} v_r.$$

The shallower checkpoints retain their unrefined states, so fusion is

$$\bar{z} = \sum_{\substack{k \in \mathcal{D} \\ k<N}} \alpha_k z^{(k-1)} + \alpha_N Z_N^{\mathrm{ref}}.$$

Substituting the two state expressions and exchanging the scan sums gives

$$\begin{aligned} \bar{z} &= \Big(\sum_{k \in \mathcal{D}} \alpha_k \Big) z^{(0)} + g_r \sum_{k \in \mathcal{D}} \alpha_k \sum_{s=1}^{k-1} u_s + \alpha_N g_r \sum_{r=1}^{R} v_r \\ &= z^{(0)} + g_r \sum_{s=1}^{N-1} \Big(\sum_{\substack{k \in \mathcal{D} \\ k>s}} \alpha_k \Big) u_s + \alpha_N g_r \sum_{r=1}^{R} v_r, \end{aligned} \tag{16}$$

because $\sum_{k \in \mathcal{D}} \alpha_k = 1$. The inner sum collects exactly those checkpoints that have consumed scan step s, yielding $\lambda_s = \sum_{k \in \mathcal{D},\, k>s} \alpha_k$ in equation 8. The post-scan vectors appear only in Z_N^{ref}, so their term retains the factor α_N.

For $1 \le s \le N-2$, $\lambda_s - \lambda_{s+1}$ equals $\alpha_{s+1} \ge 0$ if $s+1 \in \mathcal{D}$, and zero otherwise. Softmax gives $\alpha_k \ge 0$, hence for $N \ge 2$, $\lambda_1 \ge \cdots \ge \lambda_{N-1} \ge 0$. Since $1 \in \mathcal{D}$ and $N \in \mathcal{D}$, $\lambda_1 = 1 - \alpha_1$ and $\lambda_{N-1} = \alpha_N$. When $N = 1$, there are no scan updates and the ordering statement is vacuous. □

F.2 Update Magnitudes and State Dependence

Proposition 1 concerns the direct coefficient multiplying each realized recurrent update. The update magnitude itself can vary with scan depth, and each update depends on recursively accumulated context and latent states. Therefore, the monotone coefficients λ_s should not be interpreted as a monotone ordering of the end-to-end influence of individual raw patches.