AURORA-X: BUILT FOR EXTREME TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series foundation models (TSFMs) enable cross-domain forecasting, but their development as general-purpose forecasters remains constrained by under-explored training potential and limited architectural versatility. To address these challenges, we introduce Aurora-X, a billion-scale TSFM with a progressive curriculum and a unified architecture. We first use channel-independent pretraining to learn temporal patterns, then introduce cross-variable dependencies, varied context and horizon lengths, and future covariates if available during midtraining. Variable-resolution post-training further enables an adjustable temporal span per token at inference. With fixed model weights, this supports longer histories under a fixed token budget or fewer tokens for the same history, enabling test-time scaling. With a versatile architecture, Aurora-X supports cross-variable modeling, covariate conditioning, and parallel decoding of future patches for probabilistic forecasting. These are supported by a novel pattern-guided mixture-of-experts that expands model capacity through sparse activation and uses shallow patch similarities to constrain deep-layer routing, guiding expert specialization across heterogeneous time series. Furthermore, we propose an implicit quantile network head that predicts arbitrary quantiles to characterize predictive distributions, enhancing probabilistic forecasting flexibility. Comprehensive experiments on GIFT-Eval, TIME, FEV-Bench, TFB, and DAG-Bench demonstrate state-of-the-art forecasting performance against pretrained TSFMs and task-specific supervised models.

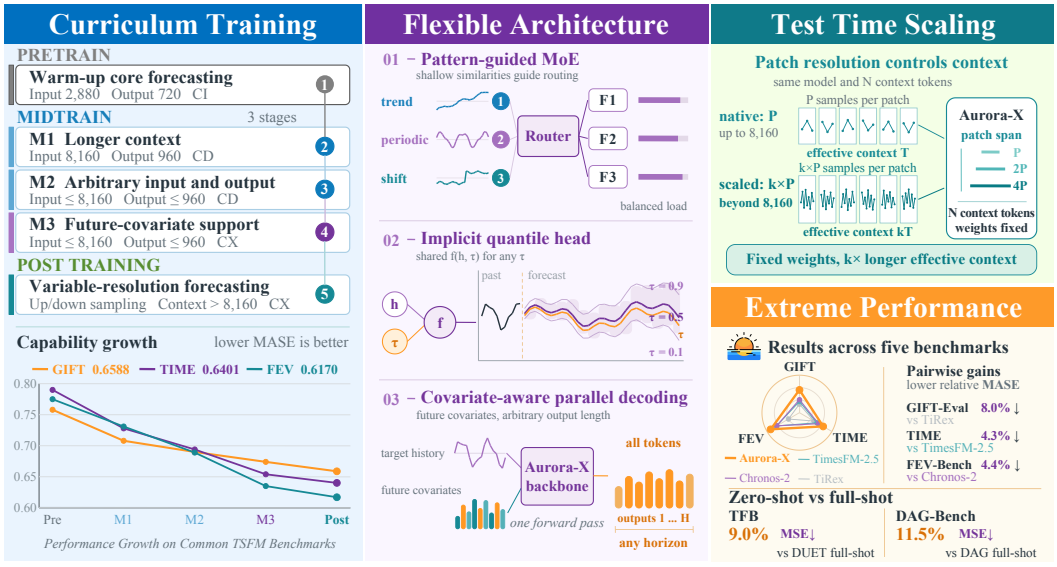


Figure 1: Overview of Aurora-X. CI, CD, and CX denote channel-independent, joint multivariate, and future-covariate training, respectively.

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.

1 INTRODUCTION

Time series foundation models (TSFMs) (Liu et al., 2024c; Google Research, 2025; Ansari et al., 2024; Wu et al., 2026) learn transferable temporal representations through large-scale pretraining on heterogeneous corpora, enabling zero-shot forecasting across datasets. Beyond zero-shot transfer, practical forecasting requires a model to integrate related variables and future covariates, characterize uncertainty flexibly, and adapt to the available history. However, existing TSFMs still face two challenges in developing and combining these capabilities.

[1] Underexplored training potential. In large language models (LLMs), pretraining builds broad linguistic competence and general knowledge, while midtraining strengthens specialized skills and extends context capacity through targeted continued pretraining (Liu et al., 2024a; Bai et al., 2023). Post-training further develops deliberative reasoning (Guo et al., 2025) and reasoning-budget control (Aggarwal & Welleck, 2025). Although staged training is emerging in TSFMs (Ansari et al., 2025; Liu et al., 2026b), how to align these stages with data characteristics, task difficulty, and target capabilities remains underexplored, particularly the potential of post-training to expand forecasting capabilities and inference-time flexibility.

[2] Limited architectural versatility. Channel independence (CI) facilitates pretraining on heterogeneous data, but CI TSFMs such as TiRex (Auer et al., 2026) and Timer-S1 (Liu et al., 2026b) do not explicitly model cross-variable dependencies or natively condition on future covariates. Chronos-2 (Ansari et al., 2025) and Falcon-X (Liu et al., 2026a) use predefined quantiles during training and inference, limiting flexibility in characterizing predictive distributions. Sundial (Liu et al., 2025) and Time-MoE (Shi et al., 2024) extend forecasts autoregressively across output blocks, requiring sequential computation and potentially accumulating errors. Overall, existing TSFM architectures have yet to combine all these capabilities within a single model, and few support test-time scaling (Table 1).

Table 1: TSFM capabilities: ✓ supported; ○ partial; ✗ unsupported. Cross-var. includes target–covariate interactions. Future cov.: future covariate values available at forecast time. Prob.: probabilistic forecasting. Cont. quant.: continuous quantile functions or sampling. Par.: parallel decoding within supported horizons (Baguan-TS: inferred). TimesFM-2.5: external regression (XReg) for covariates; blockwise decoding. TTS: test-time scaling.

Model	Cross-var.	Future cov.	Prob.	Cont. quant.	Par.	TTS
TiRex (2026)	✗	✗	✓	○	○	✓
Timer-S1 (2026)	✗	✗	✓	○	○	✗
Sundial (2025)	✗	✗	✓	✓	✗	✗
TimesFM-2.5 (2025)	○	○	✓	✓	○	✗
Time-MoE (2024)	✗	✗	✗	✗	✗	✗
Xihe (2025)	✗	✗	✓	○	✓	✗
Aurora (2026)	✗	✗	✓	✓	✓	✓
Chronos-2 (2025)	✓	✓	✓	○	✓	✗
Falcon-X (2026)	✓	✗	✓	○	✓	✗
TiRex-2 (2026)	✓	✓	✓	○	○	✗
Toto-2.0 (2026)	✓	✗	✓	○	✓	✗
Baguan-TS (2026)	✓	✓	✓	○	✓	✗
Aurora-X (Ours)	✓	✓	✓	✓	✓	✓

To address these challenges, we introduce **Aurora-X**, a billion-scale TSFM with a progressive curriculum and a versatile architecture, as shown in Figure 1. During pretraining, we adopt the CI strategy to learn general temporal patterns, then introduce joint variables, varied context and horizon lengths, and future covariates during midtraining. Through upsampling and downsampling, post-training further enables variable-resolution forecasting with an adjustable temporal span per token. With fixed model weights, this supports longer histories under a fixed token budget or fewer tokens for the same history, enabling test-time scaling.

With a versatile architecture, Aurora-X supports cross-variable and covariate conditioning, long input histories, and parallel decoding. To expand model capacity, we also propose a pattern-guided mixture of experts (MoE) with sparsely activated experts. Shallow patch similarities constrain routing in deeper layers, guiding expert specialization across heterogeneous time series. We also propose an implicit quantile network (IQN) head conditioned on continuous quantile levels. By predicting arbitrary quantiles, it characterizes predictive distributions and enhances probabilistic forecasting flexibility. Our contributions are summarized as:

❶ We propose **Aurora-X**, a billion-scale TSFM that supports cross-variable and covariate conditioning, long input histories, variable-resolution inference, and arbitrary quantile prediction within a unified architecture.

❷ We develop a progressive curriculum to learn temporal patterns during pretraining, extend cross-variable and covariate capabilities during midtraining, and enable variable-resolution inference through post-training.

❸ We combine pattern-guided MoE routing with covariate-aware parallel decoding and an implicit quantile head. Shallow patch similarities guide deep-layer expert routing, while the head supports flexible probabilistic forecasting through arbitrary quantile prediction.

❹ Evaluations against TSFMs on GIFT-Eval (Aksu et al., 2024), TIME (Qiao et al., 2026), and FEV-Bench (Shchur et al., 2025), and supervised models on TFB (Qiu et al., 2024) and DAG-Bench (Qiu et al., 2026), demonstrate state-of-the-art forecasting performance with favorable computational cost and inference latency.

2 Related Work

2.1 Time Series Foundation Models

To transfer forecasting knowledge across domains, Timer (Liu et al., 2024c), TimesFM (Das et al., 2024), and Chronos (Ansari et al., 2024) pretrain on heterogeneous corpora. Accommodating forecasting tasks with different numbers and combinations of variables motivates any-variate attention in Moirai (Woo et al., 2024). Chronos-2 (Ansari et al., 2025) and TiRex-2 (Podest et al., 2026) support multivariate forecasting with known future covariates, while Baguan-TS (Yang et al., 2026) integrates covariates into sequence-native in-context learning.

For uncertainty estimation, models offer different ways to access predictive quantiles. Moirai uses mixture distributions, Chronos-2 and Toto-2.0 (Khwaja et al., 2026) regress fixed quantile grids, and Baguan-TS derives quantiles from discretized distributions. Flow matching in Sundial (Liu et al., 2025) and Aurora (Wu et al., 2026) supports continuous generative distributions, with quantiles estimated through sampling; Aurora also conditions on text and images. Decoding imposes another constraint: TimesFM and Sundial generate successive output blocks autoregressively, while Timer-S1 (Liu et al., 2026b) retains serial hidden-state computation within a single forward pass over its supported horizon. Aurora-X directly predicts marginal quantiles at arbitrary levels over a continuous range while decoding future patches in parallel.

2.2 Training and Inference Adaptation

Expert specialization and training progression offer complementary routes to broader capabilities. Sparse experts in Time-MoE (Shi et al., 2024) and Timer-S1 scale capacity, while Pathformer (Chen et al., 2024b) and TFPS (Sun et al., 2024) guide specialization through scale-specific pathways and clustering encoded patches, respectively. Aurora-X regularizes deep-layer routing with learned shallow patch similarities, complementing load balancing. Capability development can draw on task diversity within pretraining, as in Moirai's variable context and prediction lengths, or subsequent stages: Timer-S1 uses continued pretraining and context extension, while Chronos-2 undergoes post-training for longer contexts.

Inference-time flexibility comes from computational budgets and input adaptation. Sundial varies sampling and flow-step budgets, while Baguan-TS combines retrieval with ensembles. Aurora and TiRex (Auer et al., 2026) adapt temporal resolution for periodic patterns. Aurora-X's curriculum progressively develops temporal modeling, cross-variable dependencies, and covariate conditioning; multi-scale post-training makes temporal resolution controllable at inference, allowing historical coverage to be matched to the context-token budget.

3 Aurora-X

Aurora-X is a zero-shot forecaster for cross-domain forecasting. Given histories $X \in \mathbb{R}^{C\times T}$ of target and auxiliary variables, optional future covariates if available, and quantile levels $\tau \in (0, 1)$, we predict target quantiles over the next H time steps. Figure 2 shows the full interface: patch embeddings pass through Time–Group MoE blocks to the quantile head. We pretrain on individual variables, then introduce joint variables and future covariates during midtraining.

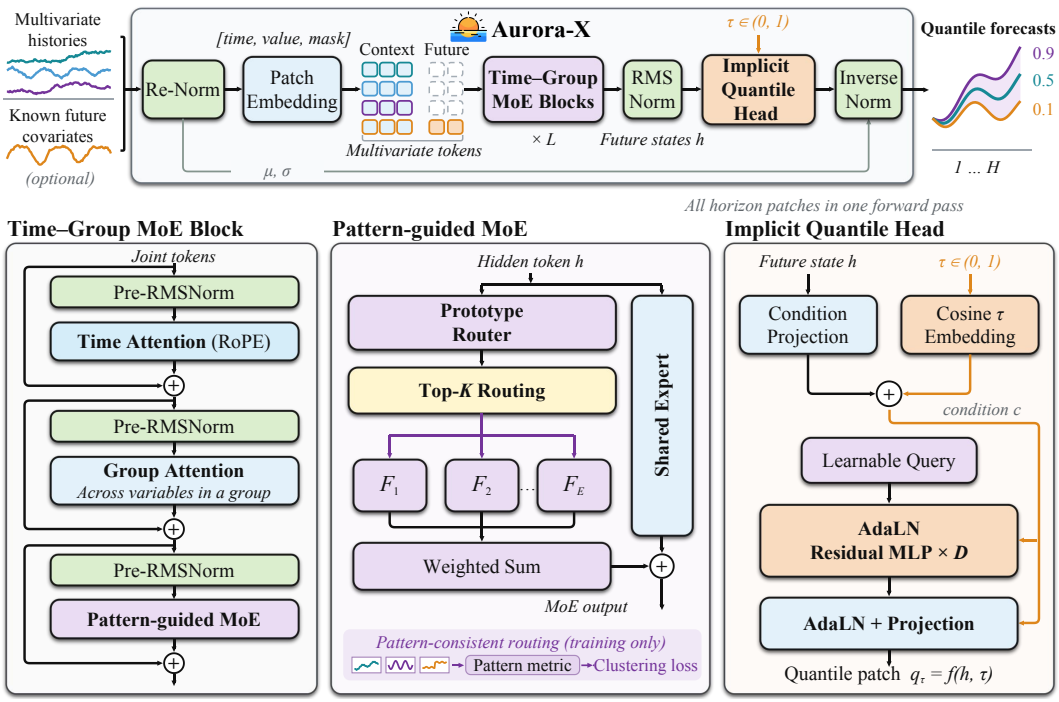


Figure 2: Overview of Aurora-X (1.05B total parameters, $L = 12$). Lower panels show a Time–Group block, its MoE sublayer, and the quantile head. Patch Embedding is fed into the blocks, whose normalized future states condition parallel quantile decoding.

3.1 Unified Backbone

Patch embedding. We first apply reversible instance normalization (RevIN) (Kim et al., 2021) using each variable's observed-history statistics, followed by arcsinh (Ansari et al., 2025) to compress extreme values. Non-overlapping patches of native length P provide $N = \lceil T/P \rceil$ context tokens for history and $M = \lceil H/P \rceil$ future tokens for prediction positions and available covariates. To support these inputs in parallel forecasting, we use a shared embedding:

$$z_{v,i}^{(0)} = \mathrm{Embed}([\mathbf{t}_i; \tilde{\mathbf{x}}_{v,i}; \mathbf{m}_{v,i}]) \in \mathbb{R}^d, \tag{1}$$

where $\mathbf{t}_i, \tilde{\mathbf{x}}_{v,i}, \mathbf{m}_{v,i} \in \mathbb{R}^P$ denote relative times, normalized values, and observation masks, and $\mathrm{Embed} : \mathbb{R}^{3P} \to \mathbb{R}^d$ is a shared residual MLP. Future targets receive zero values and masks during both training and inference, while known covariates retain observations. We exclude incomplete history patches as attention keys but keep future tokens active.

Time and group attention. We encode $Z^{(0)} \in \mathbb{R}^{C\times(N+M)\times d}$ through L Time–Group MoE blocks. Each block models temporal dependencies within variables, mixes variables in group $\mathcal{G}_g$ at aligned patch positions, and updates tokens through MoE:

$$\begin{aligned} U_{v,:,:}^{(\ell)} &= Z_{v,:,:}^{(\ell-1)} + \mathrm{Attn}_{T,\ell}^{\mathrm{RoPE}}\Big(\mathrm{RN}(Z_{v,:,:}^{(\ell-1)})\Big), \\ V_{\mathcal{G}_g,i,:}^{(\ell)} &= U_{\mathcal{G}_g,i,:}^{(\ell)} + \mathrm{Attn}_{G,\ell}\Big(\mathrm{RN}(U_{\mathcal{G}_g,i,:}^{(\ell)})\Big), \\ Z^{(\ell)} &= V^{(\ell)} + \mathrm{MoE}_\ell\Big(\mathrm{RN}(V^{(\ell)})\Big), \end{aligned} \tag{2}$$

where RN denotes RMSNorm (Zhang & Sennrich, 2019), and $U^{(\ell)}$ and $V^{(\ell)}$ are the intermediate states. Temporal attention is bidirectional over $(N+M)\times d$ states; group attention operates on $|\mathcal{G}_g|\times d$ states, both using unscaled scores. Singleton groups retain group attention during CI pretraining; midtraining enables cross-variable mixing. Finally, RMSNorm provides $h_{v,j} = \mathrm{RN}(Z^{(L)})_{v,N+j,:} \in \mathbb{R}^d$, $j = 1, \ldots, M$, to the quantile head.

3.2 PATTERN-GUIDED MIXTURE OF EXPERTS

To accommodate heterogeneous temporal patterns, we replace each Time–Group block's feed-forward sublayer with a sparsely routed MoE trained jointly with the backbone. Sparse activation expands model capacity, but load balancing alone does not guide specialization as deep temporal and cross-variable mixing can weaken pattern distinctions. We therefore guide specialization with shallow patch similarities, encouraging similar local shapes to share experts.

Prototype router and top-K routing. Each layer contains one shared expert and $E = 16$ routed experts ($F_1, \dots, F_E$ in Figure 2), with $K = 4$ activated per token. Given the MoE input $h = h_{v,i}^{(\ell)} = \mathrm{RN}(V^{(\ell)})_{v,i,:}$, we compute its affinity to each expert's learnable prototype $c_e \in \mathbb{R}^d$:

$$a_e = \frac{(W_K\, h)^\top (W_Q\, c_e)}{\sqrt{d_{\text{rout}}}}, \qquad \mathbf{s} = \mathrm{LN}_E(\mathbf{a}), \qquad \mathbf{p} = \mathrm{Softmax}(\mathbf{s}), \tag{3}$$

where $W_K, W_Q \in \mathbb{R}^{d_{\text{rout}} \times d}$ project states and prototypes, $\mathbf{a} = [a_e]_{e=1}^E$, and LN_E normalizes logits across experts without affine parameters. We select the K highest-scoring experts, denoted by $\mathcal{E}(h)$, and combine their outputs with the shared expert:

$$\mathrm{MoE}(h) = F_{\text{shared}}(h) + \sum_{e \in \mathcal{E}(h)} g_e(h) F_e(h), \qquad g_e(h) = \frac{p_e(h)}{\sum_{e' \in \mathcal{E}(h)} p_{e'}(h)}. \tag{4}$$

Here $F_e, F_{\text{shared}} : \mathbb{R}^d \to \mathbb{R}^d$ preserve the token dimension; each layer learns its own experts, prototypes, and projections.

Pattern metric and clustering loss. To preserve local pattern information in deeper layers, we learn a pattern metric from normalized context value patches during training. For patch $\tilde{\mathbf{x}}_{b,i} \in \mathbb{R}^P$ in minibatch variable row b, we compute

$$u_{b,i} = \mathrm{Normalize}(W_{\text{pat}}\, \mathrm{sg}(\tilde{\mathbf{x}}_{b,i}))\,, \qquad S_{b,ij} = \exp\left(\frac{u_{b,i}^\top u_{b,j} - 1}{\sigma_{\text{pat}}^2}\right), \tag{5}$$

where $W_{\text{pat}} \in \mathbb{R}^{d_{\text{pat}} \times P}$ is learned per layer, Normalize denotes ℓ_2 normalization, and σ_{pat} controls the similarity bandwidth. Similar projected shapes yield larger $S_{b,ij}$, providing a shallow reference for routing. The stop-gradient operator sg detaches patch inputs while keeping the projection learnable; parameter layer indices are omitted.

For context positions $i, j \le N$ within each variable row, let $D_{b,ie} = \mathbf{1}[e \in \mathcal{E}(h_{b,i}^{(\ell)})]$ indicate expert selection, with vector $\mathbf{D}_{b,i}$. We connect these discrete assignments to the pattern metric through a straight-through estimator:

$$\mathbf{A}_{b,i} = \mathrm{sg}(\mathbf{D}_{b,i}) + \mathbf{p}_{b,i} - \mathrm{sg}(\mathbf{p}_{b,i}), \qquad \Omega_{b,ij} = \frac{1}{K} \sum_{e=1}^{E} A_{b,ie} A_{b,je}. \tag{6}$$

Here $\Omega_{b,ij}$ is the fraction of selected experts shared by two patches in the forward pass, while gradients follow softmax probabilities to update the router and upstream representations. We use this overlap to weight pattern similarities and regularize the projection:

$$\mathcal{L}_{\text{pat}} = -\frac{1}{B} \sum_{b=1}^{B} \frac{\sum_{i,j=1}^{N} \Omega_{b,ij} S_{b,ij}}{\max\left(\sum_{i,j=1}^{N} \Omega_{b,ij}, \epsilon\right)}, \qquad \mathcal{R}_{\text{orth}} = \frac{\|\bar{W}_{\text{pat}}\, \bar{W}_{\text{pat}}^\top - \mathbf{I}\|_F^2}{d_{\text{pat}}^2}, \tag{7}$$

where B counts variable rows across the minibatch, $\bar{W}_{\text{pat}}$ contains row-normalized weights, $\mathbf{I}$ is the identity matrix, and $\epsilon > 0$ is a numerical floor. The normalized objective includes self-pairs. For fixed similarities, it favors sharing above the current weighted mean similarity and discourages sharing below it. Learning W_{pat} draws co-assigned patches together, while orthogonality promotes diverse projection directions. This guides specialization across heterogeneous series by linking deep expert assignments to shallow temporal patterns; Appendix C provides the analysis and proofs.

We complement pattern guidance with load balancing, $\mathcal{L}_{\text{bal}} = E \sum_e f_e \bar{p}_e$, to spread utilization across experts. Here $f_e = J^{-1} \sum_t D_{t,e}$ and $\bar{p}_e = J^{-1} \sum_t p_{t,e}$, where J counts both context and future tokens in a minibatch.

3.3 IMPLICIT QUANTILE HEAD

Probabilistic forecasting should characterize predictive distributions flexibly and efficiently. Fixed quantile heads (Ansari et al., 2025; Liu et al., 2026b) describe distributions at predefined levels, limiting direct support for customized intervals and tail queries. Sample-based forecasters (Liu et al., 2025; Wu et al., 2026) estimate these quantities from multiple generated trajectories, introducing sampling overhead. We therefore adopt an implicit quantile network (IQN) conditioned on continuous quantile levels to achieve flexible probabilistic forecasting. It directly predicts arbitrary quantiles in parallel from shared future states, enabling characterization of marginal distributions without trajectory sampling.

Quantile conditioning. Given a normalized future state $h \in \mathbb{R}^d$ from the final Time–Group MoE block (Figure 2), the head $q_\theta(h, \tau) \in \mathbb{R}^P$ predicts the conditional τ-quantile at each patch position. We combine a projection of h with a cosine embedding of the requested level $\tau \in (0, 1)$:

$$\phi(\tau) = [\cos(\pi\, n\, \tau)]_{n=0}^{R-1}, \qquad c(h, \tau) = W_h\, h + b_h + \mathrm{MLP}_\tau(\phi(\tau)) \in \mathbb{R}^d. \tag{8}$$

Here $\phi(\tau) \in \mathbb{R}^R$ represents the quantile level at multiple frequencies, and $c(h, \tau)$ integrates it with the future state. We then refine a shared learnable query $\mathbf{v}_0 \in \mathbb{R}^d$ through D residual MLP layers with adaptive layer normalization:

$$\mathbf{v}_{a+1} = \mathbf{v}_a + \alpha_a(c) \odot \mathrm{MLP}_a([1 + \gamma_a(c)] \odot \mathrm{LN}(\mathbf{v}_a) + \beta_a(c)), \tag{9}$$

where $a = 0, \ldots, D-1$, and $\beta_a(c), \gamma_a(c), \alpha_a(c) \in \mathbb{R}^d$ provide the condition-dependent shift, scale, and residual gate. This modulation incorporates the future state and requested level at every layer. A final adaptive normalization and projection yield P quantile values, with head parameters shared across variables, patches, and levels.

Quantile Optimization. During training, we independently sample Q levels from $\mathcal{U}(0, 1)$ for each target patch at every step. We supervise the corresponding predictions with the masked pinball loss:

$$\ell_{\mathrm{IQN}}(h, \tilde{\mathbf{y}}) = \frac{\sum_{q=1}^{Q} \sum_{t=1}^{P} m_t\, \rho_{\tau_q}(\tilde{y}_t - q_{\theta,t}(h, \tau_q))}{Q \max(1, \sum_{t=1}^{P} m_t)}, \qquad \rho_\tau(e) = \max\{\tau e, (\tau - 1)e\}. \tag{10}$$

Here $\tilde{y}_t$ is the normalized target and m_t indicates valid target positions. The loss weights underestimation by τ and overestimation by $1 - \tau$, guiding each prediction toward its requested conditional quantile. Averaging over target patches gives $\mathcal{L}_{\mathrm{IQN}}$; known covariates provide conditioning and are excluded from this loss. Uniform sampling approximates the integrated pinball loss over quantile levels, without an explicit non-crossing constraint.

During inference, one backbone evaluation provides the future states for parallel decoding of all patch–quantile pairs. For each target and requested level, the M predicted patches are concatenated and cropped to H steps, then transformed by $\sinh$ and inverse RevIN to obtain the forecast.

3.4 CURRICULUM AND VARIABLE-RESOLUTION POST-TRAINING

Capability-oriented curriculum. We progressively introduce richer forecasting inputs through the curriculum in Figure 1. We first pretrain on individual variables using singleton groups. During midtraining, M1 adds multivariate groups and longer contexts and horizons; M2 varies these lengths; M3 introduces future covariates if available. Finally, multi-scale post-training develops variable-resolution forecasting (Figure 3). Each stage continues the preceding checkpoint. To jointly learn quantile prediction and expert specialization, we optimize

$$\mathcal{L} = \mathcal{L}_{\mathrm{IQN}} + \sum_{\ell=1}^{L} \left[\lambda_{\mathrm{bal}} \mathcal{L}_{\mathrm{bal}}^{(\ell)} + \lambda_{\mathrm{pat}} \left(\mathcal{L}_{\mathrm{pat}}^{(\ell)} + \lambda_{\mathrm{orth}} \mathcal{R}_{\mathrm{orth}}^{(\ell)} \right) \right]. \tag{11}$$

Here λ_{bal}, λ_{pat}, and λ_{orth} control load balancing, pattern guidance, and orthogonality, respectively.

Multi-scale post-training. Pretraining and midtraining use a fixed temporal span per token (fixed resolution), so longer histories require more tokens. We therefore introduce multi-scale post-training to teach the model to forecast across resolutions, allowing longer histories within a fixed token budget or fewer tokens for the same history at inference. On a one-billion-point subset (Appendix A.4), we resample history and future separately after RevIN and arcsinh, aligning scales and block boundaries across related variables. For native patch length $P = 48$ and positive scale $k \in \mathcal{K}$, we resample blocks of integer length kP to P values; each token covers kP original points. $k < 1$ upsamples shorter spans and $k > 1$ downsamples longer spans. Let $\tilde{x}_j^{(k)} = \tilde{x}_{(j-1)kP+1:jkP}$ and $\mathcal{R}_{a\to b}$ denote linear resampling. We construct the augmented sequence as:

$$\mathcal{A}_k(\tilde{x}) = \mathrm{Concat}_j \left[\mathcal{R}_{kP\to P}\left(\tilde{x}_j^{(k)}\right) \right]. \quad (12)$$

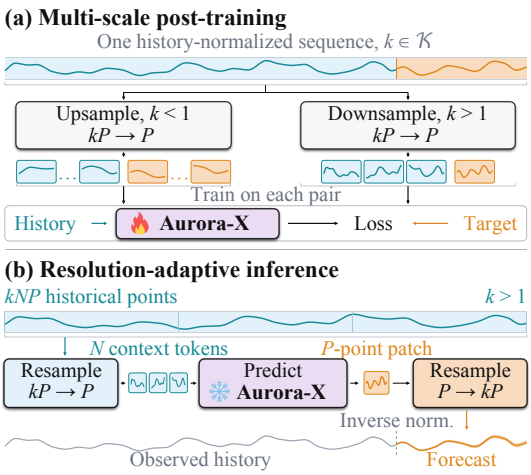


Figure 3: (a) Post-training uses resampled history–target pairs. (b) Inference extends historical coverage at a fixed context-token budget and returns forecasts to the requested resolution.

The resampled inputs contain $N_k = \lceil T/(kP) \rceil$ context tokens and $M_k = \lceil H/(kP) \rceil$ future tokens on the native P-point grid. Given their normalized future states $h_{v,j}^{(k)} \in \mathbb{R}^d$ and normalized targets $\tilde{y}_{v,1:H}$, we apply the same transformation to the targets and optimize

$$\mathcal{L}_{\mathrm{IQN}}^{\mathrm{post}}(k) = \frac{1}{|\mathcal{I}_k|} \sum_{(v,j)\in\mathcal{I}_k} \ell_{\mathrm{IQN}}\left(h_{v,j}^{(k)}, \mathcal{R}_{kP\to P}\left(\tilde{y}_{v,(j-1)kP+1:jkP}\right)\right), \quad (13)$$

where $\mathcal{I}_k$ indexes target patches (v, j), $1 \le j \le M_k$. We pad incomplete blocks and mask invalid target positions (Equation (10)). This objective replaces $\mathcal{L}_{\mathrm{IQN}}$ in Equation (11), while MoE regularization is evaluated on the augmented tokens.

Resolution-adaptive inference. During inference, we choose k to control the observations represented by each token (Figure 3(b)). We first map normalized histories and covariates with $\mathcal{R}_{kP\to P}$. After quantile decoding, we restore each length-P patch with $\mathcal{R}_{P\to kP}$, concatenate the patches, crop to H steps, and apply the inverse transformation. Model weights remain fixed, and time coordinates use the native grid. For $k > 1$, a fixed history requires fewer context tokens; at a fixed context-token budget N, historical coverage expands from NP to kNP observations. We evaluate both settings in Appendix A.11. Existing models (Wang et al., 2025a; Wu et al., 2025a; 2026) also adapt resolution; Appendices B.1–B.2 compare our resampling with projection reparameterization and LightGTS's Flex-resize (Wang et al., 2025a).

4 Experiments

We evaluate Aurora-X on five benchmarks covering general forecasting, multivariate forecasting, and forecasting with known future covariates. Figure 4 summarizes its overall performance against pretrained TSFMs and task-specific supervised models. We present the GIFT-Eval, TIME, and FEV-Bench results below, followed by analyses of expert routing, forecasting heads, and training-stage gains in Section 4.2. Comparisons on TFB and DAG-Bench are reported in Appendix A.8. Appendix A provides detailed settings, complete results, and expanded probabilistic rankings. Inference efficiency and forecasting visualizations are presented in Appendices A.12 and A.13, respectively.

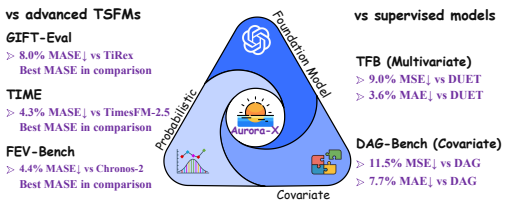


Figure 4: Forecasting performance across five benchmarks.

4.1 General Forecasting Performance

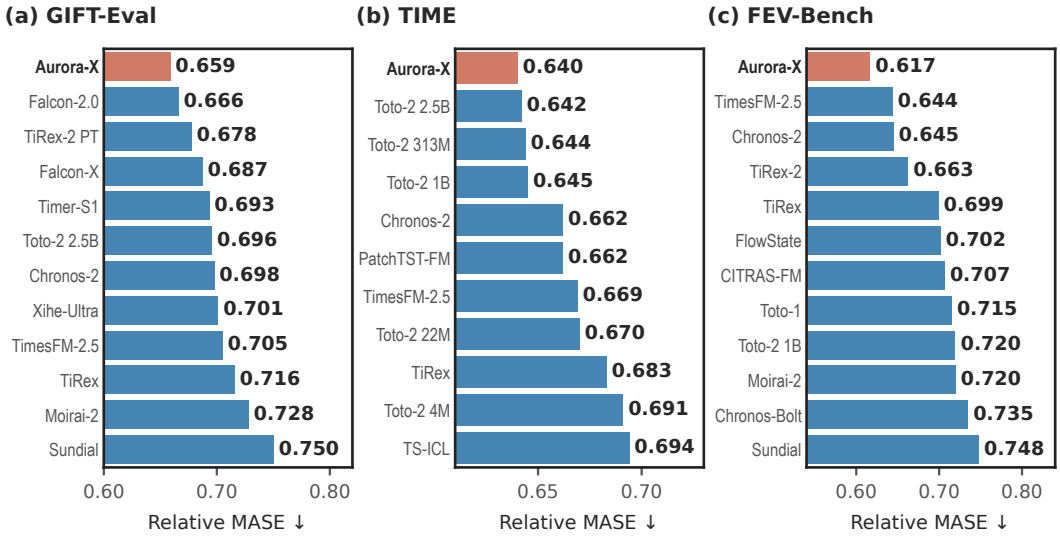


Figure 5: Relative MASE on GIFT-Eval, TIME, and FEV-Bench: geometric means of configuration-wise ratios to Seasonal Naive; lower is better. Aurora-X (1.05B total parameters) is highlighted in color. Model selection is detailed in Appendix A.2.

As shown in Figure 5, Aurora-X achieves the lowest relative MASE among the compared TSFMs on GIFT-Eval, TIME, and FEV-Bench. On GIFT-Eval, it reduces relative MASE by 1.1% compared with Falcon-2.0 and by 8.0% compared with TiRex. On TIME, the margin over Toto-2.0 (2.5B) is smaller, at 0.3%, while the reduction relative to TimesFM-2.5 is 4.3%. On FEV-Bench, Aurora-X improves over TimesFM-2.5 and Chronos-2 by 4.2% and 4.4%, respectively. These results show a consistent advantage in point forecasting across the three benchmark collections. Complementary probabilistic results are reported in Appendix A.7.

4.2 Model Analysis

The architecture comparisons use the same pretraining corpus, with all MoE variants using 16 routed experts, top-4 selection per token, and the same expert width as Aurora-X. Each comparison replaces the relevant module within the Aurora-X backbone. We examine both expert specialization and forecasting accuracy, then trace how the full curriculum develops the model's capabilities from temporal modeling to covariate conditioning and variable-resolution forecasting.

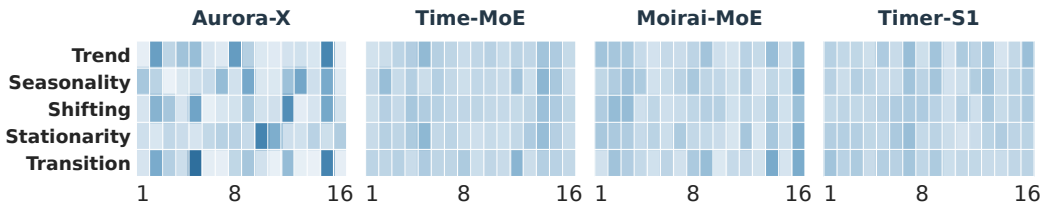


Figure 6: Expert-selection shares across five TFB-based scenarios, with 16 routed experts and top-4 selection. Columns index experts; rows sum to one. Darker shading indicates higher normalized selection frequency on a shared color scale.

MoE design. Load balancing promotes expert utilization, but specialization requires distinct routing preferences for different temporal patterns. We compare our MoE with implementations based on Time-MoE, Moirai-MoE, and Timer-S1. Following TFB's (Qiu et al., 2024) characterization, we

select datasets representing five temporal properties and inspect routing at layer 6 of the 12-layer backbone (Figure 6). Distinct rows indicate scenario-dependent expert preferences; similar rows suggest weaker differentiation across these scenarios. This probes whether input-specific routing persists after repeated representation mixing. We then compare forecasting accuracy on GIFT-Eval, TIME, and FEV-Bench (Figure 7). Appendix A.5 details dataset selection, routing-frequency aggregation, and the shared forecasting evaluation protocol for these comparisons.

Forecasting head. To examine the effect of quantile parameterization, the head comparison includes IQN, fixed quantile regression (Chronos-2), flow matching (Sundial), and a mixture distribution head (Moirai). The backbone architecture, pretraining data, and evaluation horizons are shared. Evaluation uses common quantile levels; generative heads obtain them from their predictive distributions. Figure 8 reports point accuracy from median forecasts using relative MASE against Seasonal Naive, as in Figures 5 and 7.

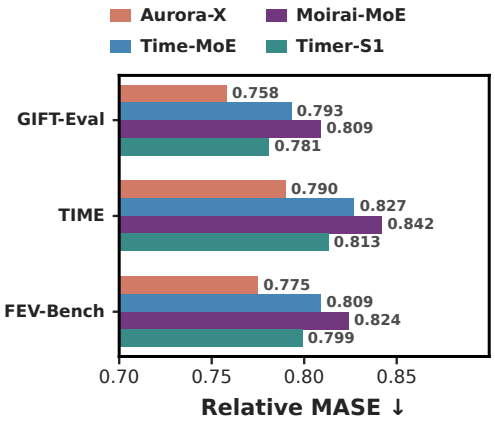


Figure 7: Relative MASE of four MoE implementations after pretraining. Lower is better.

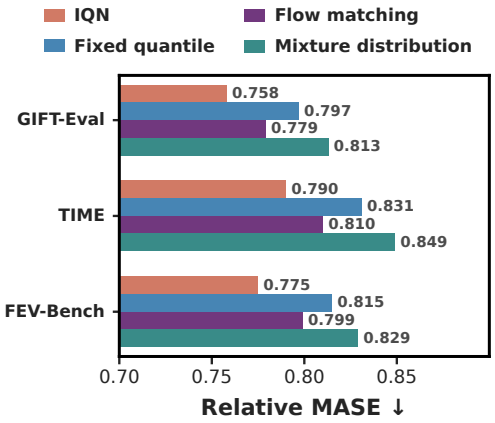


Figure 8: Relative MASE of four forecasting heads after pretraining. Lower is better.

Training-stage gains. Figure 9 follows Aurora-X through pretraining, three midtraining stages, and post-training. Relative MASE decreases at every transition, while the largest stage-wise reductions occur from Pre to M1 on GIFT-Eval and TIME, and from M2 to M3 on FEV-Bench. These trajectories show that improvements accumulate across stages, with their relative contributions varying by benchmark. Overall, relative MASE decreases by 13.1%, 19.0%, and 20.4% from Pre to Post on GIFT-Eval, TIME, and FEV-Bench, respectively. Post-training yields a further reduction of 2.1–2.8% relative to M3. Together, these results support a progressive curriculum that develops forecasting capabilities beyond the pretrained checkpoint. Appendix A.6 reports the complete stage-wise scores.

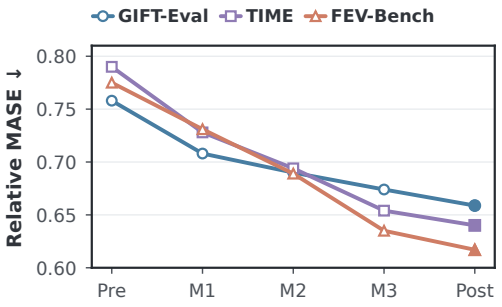


Figure 9: Relative MASE across training stages. Pre and Post denote pretraining and post-training; M1–M3 denote midtraining stages.

5 Conclusion

We introduce Aurora-X, a billion-scale time series foundation model combining pattern-guided MoE routing, an implicit quantile head, and a capability-oriented curriculum. Variable-resolution post-training enables context-token budget control and longer historical coverage. Strong performance across five forecasting benchmarks, together with favorable computational cost and inference latency, supports Aurora-X as an *out-of-the-box* tool for decision intelligence.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing. All AI-assisted text was carefully reviewed and verified by the authors. The research questions, methodology, experimental design, analysis, and scientific claims were developed independently by the authors. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work focuses on methodological research for multivariate and covariate-aware time series forecasting. Our experiments use existing benchmark datasets and pretraining corpora, without involving human subjects or the intentional collection of personally identifiable information. The proposed method is intended for general-purpose forecasting research, and we are not aware of any direct ethical risks arising from the methodology itself.

REPRODUCIBILITY STATEMENT

The model architecture and training curriculum of Aurora-X are described in Sections 3 and 3.4, respectively. Benchmark details and evaluation protocols are provided in Appendix A.1, with model configurations and training settings in Appendices A.3 and A.4. We report the configurations and evaluation protocols required to reproduce the main experiments. The source code and necessary scripts will be released.

REFERENCES

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Gift-eval: A benchmark for general time series forecasting model evaluation. *arXiv preprint arXiv:2410.10393*, 2024.

Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. Tirex: Zero-shot forecasting across long and short horizons with enhanced in-context learning. *Advances in Neural Information Processing Systems*, 38:57529–57580, 2026.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11141–11149, 2024.

Jialin Chen, Jan Eric Lenssen, Aosong Feng, Weihua Hu, Matthias Fey, Leandros Tassiulas, Jure Leskovec, and Rex Ying. From similarity to superiority: Channel clustering for time series forecasting. *arXiv preprint arXiv:2404.01340*, 2024a.

Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *ICLR*, 2024b.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *Trans. Mach. Learn. Res.*, 2023, 2023.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Google Research. TimesFM 2.5: 200M PyTorch checkpoint. Hugging Face, 2025. Model card and implementation.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yifan Hu, Peiyuan Liu, Peng Zhu, Dawei Cheng, and Tao Dai. Adaptive multi-scale decomposition framework for time series forecasting. In *AAAI*, 2025.

Emaad Khwaja, Chris Lettieri, Gerald Woo, Eden Belouadah, Marc Cenac, Guillaume Jarry, Enguerrand Paquin, Xunyi Zhao, Viktoriya Zhukov, Othmane Abou-Amal, et al. Toto 2.0: Time series forecasting enters the scaling era. *arXiv preprint arXiv:2605.20119*, 2026.

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 *ICLR*, 2021.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Yiding Liu, Yifan Hu, Hongjie Xia, Peiyuan Liu, Hongzhou Chen, Xilin Dai, Zewei Dong, and Jiang-Ming Yang. Falcon-x: A time series foundation model for heterogeneous multivariate modeling. *arXiv preprint arXiv:2605.27286*, 2026a.

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 *ICLR*, 2024b.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *International Conference on Machine Learning*, pp. 32369–32399. PMLR, 2024c.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. *arXiv preprint arXiv:2502.00816*, 2025.

Yong Liu, Xingjian Su, Shiyu Wang, Haoran Zhang, Haixuan Liu, Yuxuan Wang, Zhou Ye, Yang Xiang, Jianmin Wang, and Mingsheng Long. Timer-s1: A billion-scale time series foundation model with serial scaling. *arXiv preprint arXiv:2603.04791*, 2026b.

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

Patrick Podest, Marco Pichler, Elias Bürger, Levente Zólyomi, Bernhard Voggenberger, Wilhelm Berghammer, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. Tirex-2: Generalizing tirex to multivariate data and streaming. *arXiv preprint arXiv:2607.01204*, 2026.

Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's time: Towards the next generation of time series forecasting benchmarks. *arXiv preprint arXiv:2602.12147*, 2026.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. TFB: Towards comprehensive and fair benchmarking of time series forecasting methods. In *Proc. VLDB Endow.*, pp. 2363–2377, 2024.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *SIGKDD*, pp. 1185–1196, 2025.

Xiangfei Qiu, Yuhan Zhu, Zhengyu Li, Xingjian Wu, Bin Yang, and Jilin Hu. DAG: A dual correlation network for time series forecasting with exogenous variables. In *ICML*, 2026.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting. *arXiv preprint arXiv:2509.26468*, 2025.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv e-prints*, pp. arXiv–2409, 2024.

Yanru Sun, Zongxia Xie, Emadeldeen Eldele, Dongyue Chen, Qinghua Hu, and Min Wu. Learning pattern-specific experts for time series forecasting under patch-level distribution shift. *arXiv preprint arXiv:2410.09836*, 2024.

Yinbo Sun, Yuchen Fang, Zhibo Zhu, Jia Li, Yu Liu, Qiwen Deng, Jun Zhou, Hang Yu, Xingyu Lu, and Lintao Ma. Xihe: Scalable zero-shot time series learner via hierarchical interleaved block attention. *arXiv preprint arXiv:2510.21795*, 2025.

Yihang Wang, Yuying Qiu, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Guo Chenjuan. Lightgts: A lightweight general time series forecasting model. In *ICML*, 2025a.

Yihang Wang, Yuying Qiu, kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Guo Chenjuan. Towards a general time series forecasting model with unified representation and adaptive transfer. In *ICML*, 2025b.

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

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *ICLR*, 2023.

Xingjian Wu, Hanyin Cheng, Xiangfei Qiu, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Flame: Flow enhanced legendre memory models for general time series forecasting. *arXiv preprint arXiv:2512.14253*, 2025a.

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. In *NeurIPS*, 2025b.

Xingjian Wu, Jianxin Jin, Wanghui Qiu, Peng Chen, Yang Shu, Bin Yang, and Chenjuan Guo. Aurora: Towards universal generative multimodal time series forecasting. In *ICLR*, 2026.

Linxiao Yang, Xue Jiang, Gezheng Xu, Tian Zhou, Min Yang, ZhaoYang Zhu, Linyuan Geng, Zhipeng Zeng, Qiming Chen, Xinyue Gu, et al. Baguan-ts: A sequence-native in-context learning model for time series forecasting with covariates. *arXiv preprint arXiv:2603.17439*, 2026.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, volume 37, pp. 11121–11128, 2023.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *ICLR*, 2022.

A Experimental Settings and Additional Results

A.1 Benchmarks and Evaluation Protocol

We evaluate Aurora-X on five benchmarks covering general forecasting across domains, multivariate prediction, and forecasting with future covariates. Together, they assess transfer across temporal patterns and the ability to exploit different forms of predictive information.

GIFT-Eval (Aksu et al., 2024) evaluates general forecasting across diverse domains, sampling frequencies, and prediction lengths. We use its 97 configurations, whose short-, medium-, and long-term tasks assess forecasting under heterogeneous temporal scales.

TIME (Qiao et al., 2026) is a task-centric benchmark for zero-shot forecasting on newly collected data. It contains 98 tasks from 50 datasets across eight domains, with task configurations designed around application requirements and variate predictability. Its feature-based analyses further characterize performance across temporal patterns.

FEV-Bench (Shchur et al., 2025) comprises 100 forecasting tasks from 96 datasets across seven domains. It includes univariate and multivariate targets, static attributes, past-only dynamic covariates, and known future covariates, providing varied input structures for assessing general forecasting models.

TFB (Qiu et al., 2024) provides a unified evaluation framework for univariate and multivariate forecasting across diverse temporal characteristics. We use eight multivariate datasets: ETTh1, ETTh2, ETTm1, ETTm2, Weather, Solar, Electricity, and Traffic, with prediction lengths of 96, 192, 336, and 720.

DAG-Bench (Qiu et al., 2026) refers to the twelve real-world covariate-forecasting datasets evaluated in DAG. They cover electricity prices, energy generation, reservoir water levels, and wind power. Forecast horizons are 24 and 360 for ten datasets, and 10 and 30 for Colbun and Rapel. These tasks evaluate how models use future exogenous information to improve target forecasts.

Evaluation protocol. We follow the benchmark-defined forecasting tasks, data splits, prediction horizons, and metric definitions. Each benchmark is evaluated under its corresponding input setting, including the availability of future covariates.

Reported aggregation and input configurations. Relative MASE is the geometric mean of configuration-wise ratios to Seasonal Naive. FEV-Bench uses the recorded task-level ratios directly, without leaderboard clipping, failure imputation, or training-overlap substitutions. TFB and DAG-Bench report arithmetic horizon averages of MSE and MAE on normalized series; aggregate comparisons weight datasets equally. Means are computed from the reported three-decimal horizon-specific scores, and percentage improvements are calculated before display rounding.

On TFB, Aurora-X uses 2,880 history points and 48 points per patch, except on Solar, where the patch span is 144. The eight baselines taken from the SEER table use 96 history points; other baselines retain their published configurations. GIFT-Eval and TIME use dataset-specific inference spans selected with benchmark feedback. On DAG-Bench, models receive future covariates through the benchmark's input adapters.

A.2 Baselines and Result Sources

Time series foundation models. We compare Aurora-X with pretrained forecasters spanning different architectures and probabilistic prediction strategies. Timer-S1 (Liu et al., 2026b) combines sparse MoE blocks with serial-token prediction. Chronos-2 (Ansari et al., 2025) uses group attention to share information among related series, target variables, and covariates. TiRex (Auer et al., 2026) employs an xLSTM backbone for recurrent state tracking, while TiRex-2 (Podest et al., 2026) extends this approach to multivariate inputs, future covariates, and streaming inference. Falcon-X (Liu et al., 2026a) models heterogeneous variables through a shared latent prototype space. The Xihe family (Sun et al., 2025) captures local and global temporal dependencies with hierarchical interleaved block attention; our comparison includes Xihe-Ultra. Sundial (Liu et al., 2025) learns predictive distributions through flow matching, and Toto-2.0 (Khwaja et al., 2026) provides fore-

casting models at multiple parameter scales. We also include TimesFM-2.5, Moirai-2.0, Falcon-2.0, FlowState, CITRAS-FM, Chronos-Bolt, Toto-1.0, PatchTST-FM-Extended, and TS-ICL. Evaluated versions, sizes, and checkpoint settings are identified in the results.

Supervised multivariate forecasting. The TFB comparison covers fourteen supervised baselines. DUET (Qiu et al., 2025) combines temporal and channel clustering, while AMD (Hu et al., 2025) models temporal and channel dependencies through adaptive multiscale decomposition. SRSNet (Wu et al., 2025b) adaptively selects and reassembles informative temporal patches. PatchTST (Nie et al., 2023) uses channel-independent patch tokens; iTransformer (Liu et al., 2024b) attends over variate tokens, and Crossformer (Zhang & Yan, 2022) separates temporal and cross-dimension attention. MSGNet (Cai et al., 2024) learns inter-series correlations at multiple temporal scales using adaptive graph convolution. TimesNet (Wu et al., 2023) represents periodic variation in two-dimensional tensors, and DLinear (Zeng et al., 2023) provides a linear forecasting baseline. TimeKAN, TimePro, xPatch, Amplifier, and Fredformer complete the comparison.

Forecasting with covariates. DAG-Bench includes ten supervised baselines. DAG (Qiu et al., 2026) models temporal and channel relationships between endogenous and exogenous variables through dual correlation networks. TimeXer (Wang et al., 2024) combines patch-wise self-attention with variate-wise cross-attention to incorporate exogenous information. TiDE (Das et al., 2023) uses an MLP encoder–decoder to model temporal dependencies and covariates. We additionally compare with GCGNet, TFT, CrossLinear, DUET, Amplifier, TimeKAN, and PatchTST under the benchmark's covariate-input setting.

Model selection and result sources. The main MASE figure compares individual TSFMs and excludes benchmark-specific fine-tuning, ensembles, and tabular foundation-model adaptations. Its GIFT-Eval panel requires complete recorded coverage and no disclosed test overlap; expanded rankings include additional checkpoints, with fine-tuned variants marked explicitly. TIME includes the reported Toto-2.0 sizes and PatchTST-FM-Extended; PatchTST-FM-R1 and OmniScient are excluded. For duplicated TFB baselines, we use the SEER table for DUET, Fredformer, iTransformer, PatchTST, SRSNet, xPatch, Amplifier, and DLinear; the other six methods use the CCD table. CCD and SEER are excluded. DAG-Bench retains all ten published baselines. Each table ranks its displayed models separately; red boldface and blue underlining mark the best and second-best distinct scores, including all ties at three-decimal precision.

A.3 Model Configuration

Aurora-X has approximately one billion total parameters. Its backbone comprises 12 Time–Group blocks with a hidden width of 768 and a native patch length of 48. Each MoE layer combines one shared expert with 16 routed experts, selecting four routed experts per token. Group attention, future time encoding, and the arcsinh transformation are enabled. Table 2 summarizes the model configuration.

For IQN training, each target patch and its conditioning state are replicated five times, with 20 independently sampled quantile levels per replica. This gives $Q = 100$ quantile samples per original target patch in Equation (10); inference can query arbitrary continuous quantile levels. The regularization coefficients follow the nested weighting in Equation (11), with the orthogonality penalty multiplied by both λ_{pat} and λ_{orth}. The router uses a Switch-style auxiliary load-balancing loss; the separate expert-bias update rate is zero.

The serialized configuration records `float32` as its dtype and Transformers version 4.50.1.

A.4 Training Data and Five-Stage Curriculum

Training corpora. Pretraining and midtraining use the open-source TimeBench corpus adopted by Sundial (Liu et al., 2025) and Timer-S1 (Liu et al., 2026b). It brings together heterogeneous time series from domains including meteorology, healthcare, finance, and the Internet of Things. For post-training, we construct a subset of one billion original time points from the pretraining corpora used by LightGTS (Wang et al., 2025a) and ROSE (Wang et al., 2025b). This size refers to the source observations before multi-scale augmentation.

Table 2: Aurora-X model configuration. Parameter count denotes total model size; expert counts are per MoE layer.

Component	Setting	Value
Model scale	Total parameters	1B
Backbone	Time–Group blocks	12
	Hidden width	768
	Attention heads	12
	Feed-forward hidden width	3,072
	Native patch length	48
	Backbone and expert activation	ReLU
	Dropout rate	0.1
	Time-encoding scale	8,160
	RoPE base	10,000
	Initialization factor	0.05
Mixture of experts	Routed experts	16
	Shared experts	1
	Active routed experts per token	4
	Expert hidden width	3,072
	Router dimension	768
	Shallow-pattern dimension	32
	Load balancing	Switch-style
Implicit quantile head	Residual blocks	4
	Cosine features	128
	Quantile samples per replica	20
	Target/condition replication factor	5
	Head activation	SiLU
MoE regularization	Pattern-loss coefficient λ_{pat}	0.01
	Pattern-similarity bandwidth σ_{pat}	1.0
	Relative orthogonality coefficient λ_{orth}	0.1
	Load-balancing coefficient λ_{bal}	0.001
	Expert-bias update rate	0.0

Training resources and schedule. We train Aurora-X using 64 NVIDIA H800 GPUs and AdamW with an initial learning rate of 5×10^{-5}, cosine annealing, and FP32 precision. We use a per-GPU batch size of 128 and accumulate gradients over three steps, yielding an effective batch size of 24,576. The curriculum consists of pretraining (Pre), three midtraining stages (M1–M3), and post-training (Post), with each stage continuing from the preceding checkpoint. Table 3 summarizes the 160K optimization steps across all five stages.

Table 3: Optimization steps across the five training stages of Aurora-X. K denotes 1,000 steps.

Stage	Training focus	Steps
Pre	Channel-independent temporal modeling	50K
M1	Cross-variable modeling with extended contexts and horizons	25K
M2	Adaptation to varied context lengths and forecast horizons	25K
M3	Known future covariate conditioning	50K
Post	Variable-resolution forecasting	10K
Total		**160K**

Pre: temporal pattern modeling. We begin with channel-independent forecasting on TimeBench. Each variable is modeled through its own historical observations, allowing the shared backbone to learn temporal patterns across heterogeneous sources before introducing cross-variable interactions.

M1: cross-variable dependency modeling. We introduce jointly modeled variables and extend the context and forecast horizon. Group attention can then combine each target's temporal history with information from related variables, developing multivariate forecasting capabilities.

M2: context and horizon adaptation. We vary historical context lengths and forecast horizons during training. This exposes the same model to different amounts of conditioning information and prediction demands, supporting flexible forecasting configurations.

M3: future covariate conditioning. We introduce known future covariates alongside target histories. Available covariate values condition the forecast, while prediction losses are evaluated only on target variables. This stage teaches the model to use auxiliary information extending into the forecast horizon.

Post: variable-resolution forecasting. We augment the one-billion-point subset by enumerating temporal spans and resampling each block to the native patch length. Histories, covariates, and targets undergo aligned transformations, with supervision evaluated on the resampled targets. This stage develops the ability to adjust temporal resolution at inference, controlling historical coverage and context-token usage with fixed model weights.

Pre and M1–M3 use the quantile objective and MoE regularizers in Equation (11). Post uses the resampled-target objective in Equation (13) with the same regularizers. The construction of paired resolution augmentations and their alignment with inference are detailed in Appendix B.

A.5 Architecture Comparison Protocol

Shared pretraining setting. The architecture studies compare module implementations within Aurora-X, using our pretraining corpus. The protocol matches training exposure, backbone dimensions, and evaluation configurations. All four MoE variants use 16 routed experts with top-4 selection per token and expert width matched to Aurora-X; shared-expert configurations and module-specific objectives are recorded for each variant. The head comparison retains the Aurora-X MoE and changes only the forecasting head and its associated objective. These comparisons use the pretraining checkpoint, while the training-stage analysis follows the full curriculum.

Routing across distinct inputs. We follow the characteristic-based selection in TFB (Qiu et al., 2024), Section 5.2.3, which identifies a representative dataset with the highest score for each characteristic. Table 4 lists the five temporal scenarios used here. These descriptors identify prominent properties, rather than mutually exclusive dataset classes. TFB's transition score describes symbolic transition regularity; shifting captures changes in the value distribution.

Table 4: Routing scenarios following TFB's characteristic analysis. Each dataset is TFB's highest-scoring representative for the corresponding characteristic.

Characteristic	Dataset
Trend	FRED-MD
Seasonality	Electricity
Shifting	NYSE
Stationarity	Solar
Transition	PEMS08

To isolate temporal routing behavior, we sample channel-wise context windows at the native patch resolution from held-out data. We fix window length and sample equal numbers of valid context patches per dataset, selecting channels and windows uniformly. All variants receive identical probe windows, and routing is evaluated at layer 6 of the 12-layer backbone; future-token routing is excluded. This intermediate-layer probe examines whether distinct input distributions retain different expert preferences after multiple representation transformations. Neither window selection nor dataset assignment depends on a learned router or its shallow-pattern clusters. Let $c_{s,e}$ count selections of expert e in probe window s. For a dataset group $\mathcal{G}$, the heatmap reports

$$P_{\mathcal{G},e} = \frac{1}{|\mathcal{G}|} \sum_{s \in \mathcal{G}} \frac{c_{s,e}}{\sum_{e'} c_{s,e'}}.$$

Each sequence receives equal weight; rows sum to one, and darker shading indicates higher normalized expert-selection frequency. Rows describe input-dependent expert preferences; their group-balanced average describes overall utilization. Expert indices are local to each variant, and shared experts are excluded.

Forecasting evaluation. All variants use common benchmark configurations and relative MASE aggregation. Head comparisons use a common evaluation quantile grid containing the median. IQN and the fixed quantile head predict these levels directly; distributional heads estimate them from their predictive distributions, with the extraction procedure and inference budget recorded per run.

Routing across depth. Figure 10 presents routing profiles across all 12 layers, with layer 6 reproducing the profiles in Figure 6. The layer-wise panels compare hotspot locations, scenario overlap, and background utilization across depth. Within these panels, Aurora-X has distinct scenario profiles at every layer. The other three implementations show clear scenario differences in layers 1–3; these differences weaken from layer 4 onward. Their deeper profiles become similar across scenarios within each layer, while the preferred experts can still differ between layers. All panels use a common color scale. Expert indices are local to each layer and variant, so matching column positions across layers do not imply the same learned expert function.



Figure 10: Routing profiles across all 12 layers. Each layer contains four model panels, with five scenario rows and 16 expert columns per panel. Rows sum to one; darker shading indicates higher selection frequency on a shared color scale. Layer 6 matches Figure 6.

A.6 Performance across Training Stages

Figure 11 tracks forecasting accuracy as Aurora-X progresses through the curriculum. Relative MASE decreases at every stage on all three benchmarks. From pretraining to post-training, the reductions are 13.1% on GIFT-Eval, 19.0% on TIME, and 20.4% on FEV-Bench. The largest absolute decrease occurs from Pre to M1 on GIFT-Eval and TIME, and from M2 to M3 on FEV-Bench. Post-training further reduces relative MASE by 2.3%, 2.1%, and 2.8% compared with M3, respectively, showing continued gains after midtraining.

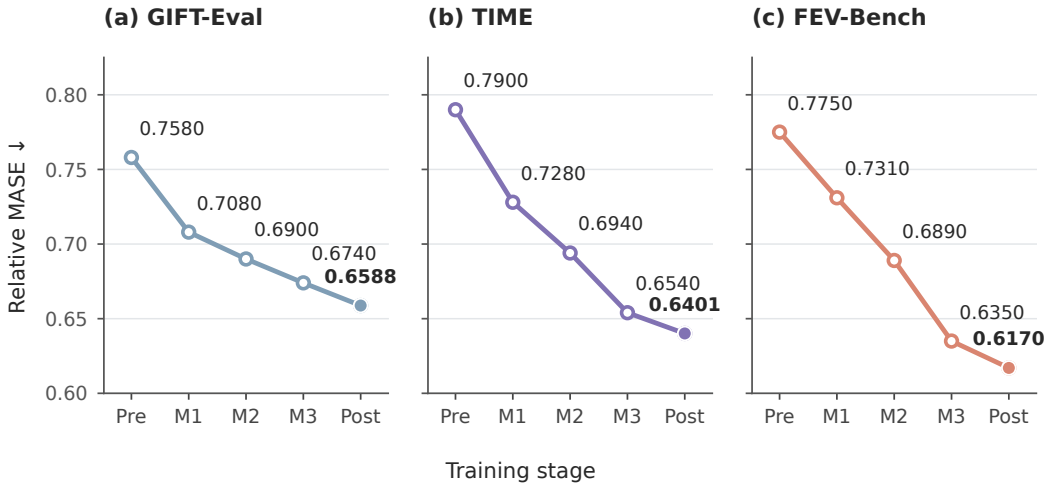


Figure 11: Relative MASE across successive Aurora-X training stages; lower is better. Pre and Post denote pretraining and post-training, while M1–M3 denote successive midtraining stages. All panels use the same vertical scale.

A.7 Expanded Probabilistic Forecasting Rankings

Figures 12 and 13 complement the main MASE comparison with expanded probabilistic rankings. GIFT-Eval uses weighted quantile loss (WQL), while FEV-Bench uses scaled quantile loss (SQL), normalized by in-sample seasonal differences. We report geometric means of model-to-Seasonal-Naive metric ratios; lower values are better. Models must have complete recorded MASE coverage and an available probabilistic score.

The GIFT-Eval ranking includes 43 models and checkpoints, with the two fine-tuned variants explicitly distinguished. Aurora-X obtains relative WQL of 0.4665, compared with 0.4669 for TiRex-2 (PT) and 0.4634 for the fine-tuned Toto-2.0 (2.5B) checkpoint. On FEV-Bench, Aurora-X achieves relative SQL of 0.5168, the lowest among the twelve displayed models. Different model sizes and training settings are listed separately.

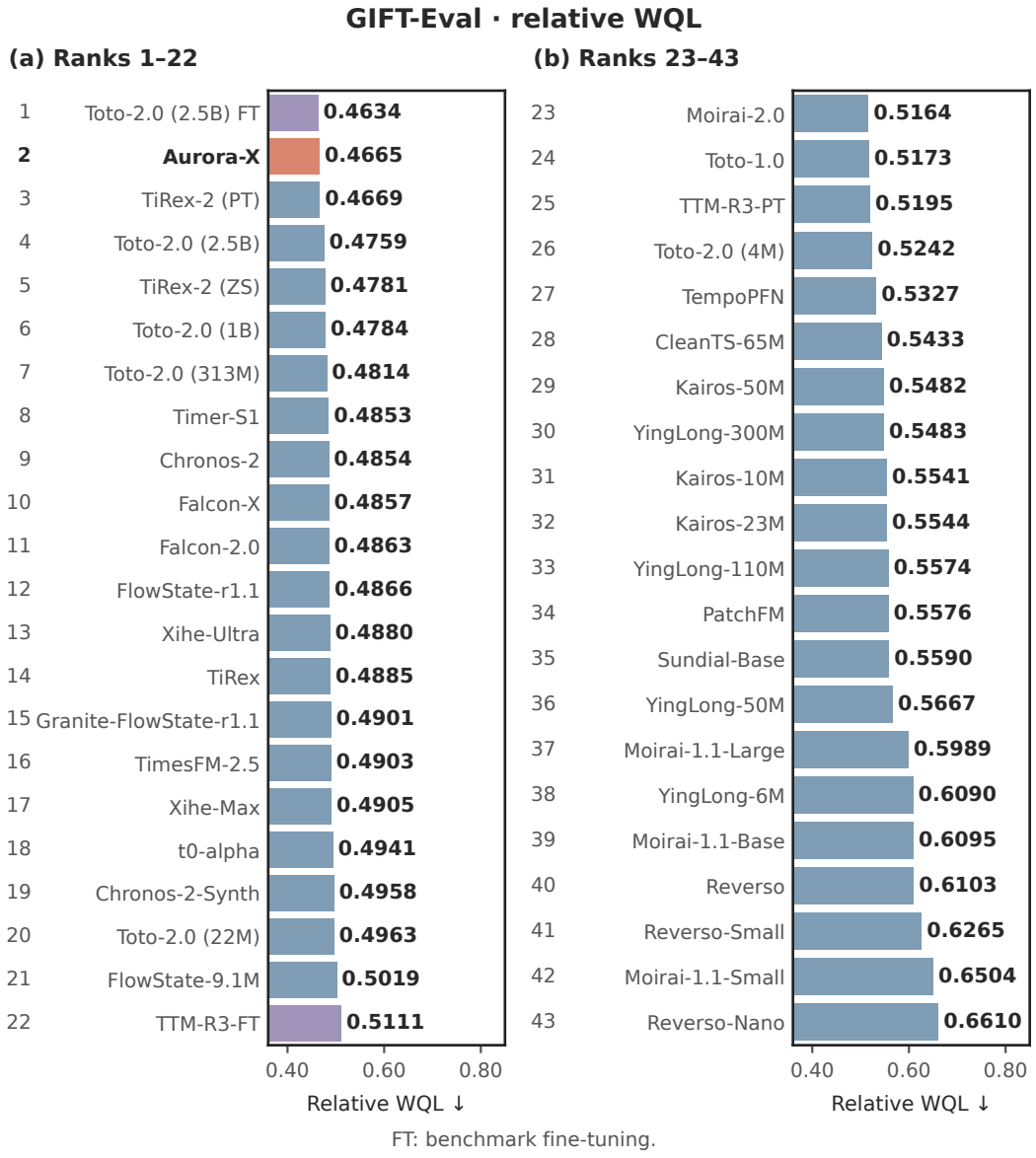


Figure 12: Expanded WQL ranking on GIFT-Eval. The two panels show a single ranking of 43 models and checkpoints. Lavender bars and FT labels identify fine-tuned variants. Scores use four decimal places; lower is better.

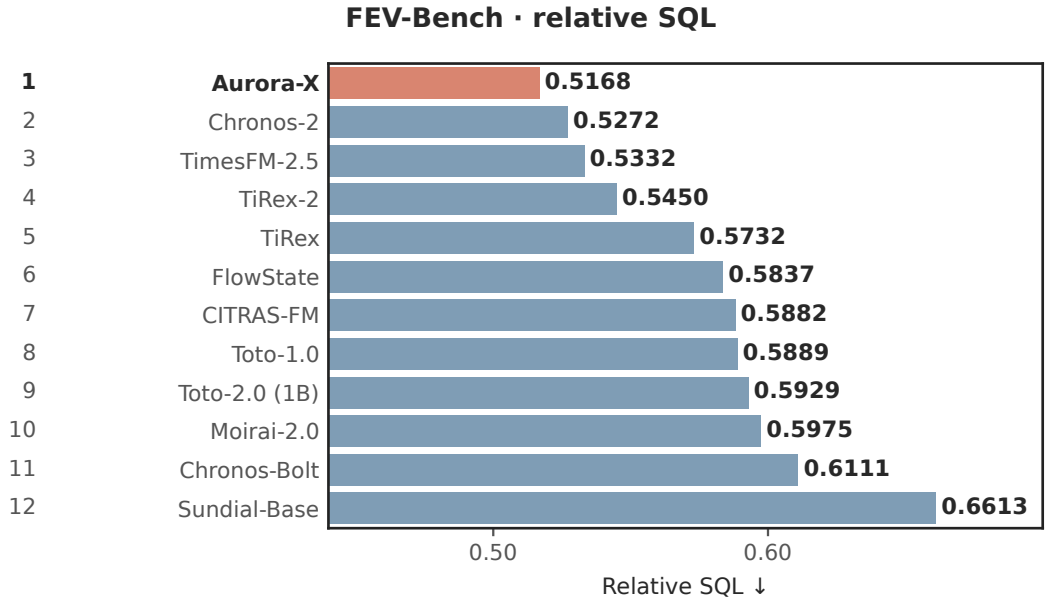


Figure 13: Relative SQL of twelve TSFMs on FEV-Bench. Scores use four decimal places; lower is better.

A.8 MULTIVARIATE AND COVARIATE FORECASTING

Tables 5 and 6 report multivariate forecasting on TFB and forecasting with known future covariates on DAG-Bench, using task-specific supervised models as baselines. On TFB, Aurora-X reduces average MSE by 9.0% and MAE by 3.6% relative to DUET. It obtains the lowest horizon-averaged MSE on seven of the eight datasets and the lowest MAE on five. On DAG-Bench, Aurora-X reduces average MSE by 11.5% and MAE by 7.7% relative to DAG, ranking first in MSE on nine of the twelve datasets and in MAE on six. Its performance is particularly strong on the four SDWPF subsets, where it achieves the lowest values for both metrics. These rankings use the complete baseline sets in Appendices A.9 and A.10. The results demonstrate that the shared forecasting model remains competitive with supervised predictors in both settings.

Table 5: Aurora-X is evaluated against recent advanced supervised baselines on TFB. MSE and MAE are averaged over prediction horizons of 96, 192, 336, and 720 steps. Lower values are better. Red boldface and blue underlining denote the best and second-best distinct values within this table, including all ties at the displayed precision. Differences in context settings are detailed in Appendix A.1, and full results are provided in Appendix A.9.

Dataset	Aurora-X		DUET		TimeKAN		AMD		TimePro		SRSNet		xPatch		Amplifier	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.390**	**0.411**	0.443	0.436	0.431	0.429	0.439	0.430	0.438	0.438	0.442	0.434	0.444	0.429	0.449	0.437
ETTh2	**0.353**	**0.380**	0.372	0.398	0.391	0.412	0.371	0.398	0.377	0.404	0.376	0.402	0.369	0.392	0.389	0.411
ETTm1	**0.335**	**0.364**	0.390	0.393	0.384	0.401	0.389	0.396	0.391	0.400	0.385	0.395	0.387	0.383	0.383	0.396
ETTm2	**0.268**	**0.313**	0.280	0.324	0.282	0.330	0.278	0.323	0.281	0.326	0.284	0.330	0.280	0.319	0.280	0.326
Weather	**0.211**	**0.245**	0.251	0.273	0.245	0.273	0.254	0.280	0.252	0.276	0.250	0.277	0.247	0.266	0.247	0.275
Solar	**0.222**	0.242	0.237	**0.233**	0.255	0.282	0.253	0.281	0.233	0.267	0.250	0.281	0.277	0.268	**0.222**	0.256
Electricity	0.170	0.266	0.172	**0.259**	0.197	0.286	0.187	0.281	**0.169**	0.263	0.189	0.276	0.184	0.266	0.174	0.268
Traffic	**0.413**	0.270	0.451	**0.269**	0.455	0.318	0.500	0.324	0.447	0.302	0.495	0.307	0.500	0.283	0.485	0.315

Table 6: Aurora-X is compared with supervised covariate-aware baselines on DAG-Bench. MSE and MAE are averaged over prediction horizons of 24 and 360 steps, or 10 and 30 steps for Colbun and Rapel. Lower values are better. Red boldface and blue underlining denote the best and second-best distinct values within this table, including all ties at the displayed precision. Full results for each horizon are reported in Appendix A.10.

Dataset	Aurora-X		DAG		GCGNet		TimeXer		TFT		DUET		Amplifier		TimeKAN	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
NP	**0.359**	0.346	0.362	**0.344**	0.370	0.348	0.418	0.371	0.379	0.375	0.411	0.408	0.420	0.419	0.406	0.420
PJM	0.112	0.204	**0.094**	**0.181**	0.095	0.187	0.108	0.199	0.114	0.207	0.102	0.197	0.137	0.247	0.139	0.263
BE	**0.413**	0.283	0.423	**0.280**	0.431	0.294	0.452	0.290	0.454	0.291	0.515	0.354	0.559	0.413	0.548	0.407
FR	**0.411**	0.225	0.414	**0.220**	0.415	0.234	0.428	0.241	0.504	0.257	0.496	0.327	0.554	0.408	0.548	0.374
DE	0.427	0.409	**0.370**	**0.370**	0.402	0.389	0.475	0.418	0.490	0.446	0.483	0.430	0.473	0.441	0.473	0.446
Energy	0.197	0.343	**0.124**	**0.268**	0.132	0.278	0.163	0.315	0.130	0.283	0.203	0.368	0.233	0.389	0.219	0.381
Sdwpfm1	**0.330**	**0.388**	0.423	0.461	0.424	0.457	0.702	0.609	0.482	0.475	0.599	0.571	0.437	0.490	0.447	0.535
Sdwpfm2	**0.340**	**0.399**	0.478	0.485	0.475	0.486	0.803	0.653	0.476	0.489	0.515	0.490	0.491	0.513	0.497	0.564
Sdwpfh1	**0.312**	**0.366**	0.449	0.486	0.450	0.500	0.746	0.644	0.479	0.492	0.539	0.516	0.537	0.598	0.577	0.638
Sdwpfh2	**0.312**	**0.383**	0.523	0.530	0.520	0.536	0.891	0.719	0.566	0.521	0.647	0.566	0.521	0.581	0.647	0.672
Colbun	**0.093**	**0.145**	0.098	0.155	0.107	0.176	0.145	0.236	0.238	0.298	0.198	0.266	0.173	0.246	0.128	0.175
Rapel	**0.226**	**0.279**	0.230	0.306	0.306	0.307	0.344	0.362	0.305	0.334	0.270	0.326	0.257	0.322	0.250	0.311

A.9 Complete Multivariate Forecasting Results

Tables 7 and 8 report all TFB results at horizons 96, 192, 336, and 720. Table 5 summarizes seven recent advanced supervised baselines; the full tables include all fourteen, divided into two groups with Aurora-X first in each. Each table ranks its displayed models separately, including all ties at the reported precision. Both follow the source selection described above.

Table 7: Full TFB results, model group 1. Lower is better; red bold and blue underlining indicate the best and second-best distinct values within this table, including all ties at the displayed precision.

Dataset	*H*	**Aurora-X**		**DUET**		**TimeKAN**		**AMD**		**TimePro**		**SRSNet**		**xPatch**		**Amplifier**	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.362**	**0.390**	0.377	0.393	0.374	0.391	0.375	0.392	0.375	0.398	0.383	0.395	0.378	**0.390**	0.376	0.393
	192	**0.392**	**0.409**	0.429	0.425	0.421	0.421	0.430	0.422	0.427	0.429	0.433	0.422	0.433	0.420	0.442	0.430
	336	**0.404**	**0.416**	0.471	0.446	0.464	0.440	0.471	0.443	0.472	0.450	0.476	0.446	0.484	0.445	0.478	0.446
	720	**0.401**	**0.427**	0.496	0.480	0.466	0.462	0.478	0.464	0.476	0.474	0.474	0.471	0.480	0.462	0.501	0.479
ETTh2	96	0.291	0.334	0.296	0.345	0.293	0.343	**0.287**	0.338	0.293	0.345	0.296	0.345	**0.287**	**0.332**	0.298	0.347
	192	**0.349**	**0.373**	0.368	0.389	0.375	0.396	0.367	0.388	0.367	0.394	0.369	0.392	0.360	0.382	0.378	0.401
	336	**0.375**	**0.394**	0.411	0.422	0.429	0.441	0.410	0.424	0.419	0.431	0.413	0.425	0.417	0.421	0.428	0.437
	720	**0.396**	**0.420**	0.412	0.434	0.466	0.468	0.421	0.440	0.427	0.445	0.425	0.444	0.412	0.433	0.452	0.460
ETTm1	96	**0.285**	**0.332**	0.324	0.354	0.327	0.365	0.327	0.361	0.326	0.361	0.319	0.358	0.316	0.343	0.318	0.356
	192	**0.321**	**0.354**	0.369	0.379	0.363	0.387	0.366	0.383	0.367	0.383	0.359	0.381	0.369	0.369	0.362	0.381
	336	**0.347**	**0.371**	0.404	0.402	0.389	0.407	0.398	0.404	0.402	0.409	0.391	0.404	0.401	0.392	0.393	0.404
	720	**0.387**	**0.398**	0.463	0.437	0.457	0.445	0.464	0.437	0.469	0.446	0.470	0.436	0.461	0.429	0.460	0.442
ETTm2	96	**0.174**	**0.248**	**0.174**	0.255	0.178	0.262	0.176	0.259	0.178	0.260	0.181	0.267	**0.174**	0.252	0.178	0.261
	192	**0.235**	**0.291**	0.243	0.302	0.244	0.308	0.242	0.302	0.242	0.303	0.243	0.306	0.240	0.297	0.243	0.303
	336	**0.290**	**0.329**	0.304	0.341	0.305	0.346	0.298	0.337	0.303	0.342	0.306	0.346	0.302	0.335	0.305	0.344
	720	**0.371**	**0.384**	0.399	0.397	0.402	0.404	0.396	0.394	0.400	0.399	0.407	0.399	0.403	0.393	0.393	0.397
Weather	96	**0.148**	**0.189**	0.163	0.202	0.164	0.210	0.174	0.221	0.166	0.207	0.167	0.214	0.166	0.202	0.165	0.210
	192	**0.185**	**0.226**	0.218	0.252	0.209	0.250	0.219	0.259	0.216	0.254	0.215	0.255	0.210	0.242	0.212	0.253
	336	**0.226**	**0.260**	0.274	0.294	0.264	0.290	0.273	0.296	0.273	0.296	0.270	0.294	0.267	0.284	0.267	0.293
	720	**0.286**	**0.304**	0.349	0.343	0.343	0.342	0.349	0.345	0.351	0.346	0.346	0.344	0.344	0.335	0.344	0.342
Solar	96	0.201	0.225	0.200	**0.207**	0.228	0.254	0.209	0.241	0.196	0.237	0.216	0.258	0.234	0.245	**0.186**	0.232
	192	**0.223**	0.241	0.228	**0.233**	0.241	0.284	0.235	0.260	0.231	0.263	0.247	0.280	0.265	0.262	0.231	0.264
	336	0.238	0.252	0.262	**0.244**	0.273	0.292	0.294	0.316	0.250	0.281	0.268	0.294	0.301	0.280	**0.234**	0.263
	720	**0.225**	**0.249**	0.258	**0.249**	0.279	0.297	0.275	0.308	0.253	0.285	0.268	0.290	0.308	0.284	0.238	0.265
Electricity	96	0.146	0.244	0.145	**0.233**	0.174	0.266	0.147	0.251	**0.139**	0.234	0.161	0.252	0.160	0.244	0.149	0.245
	192	0.159	0.255	0.163	**0.248**	0.182	0.273	0.176	0.262	**0.156**	0.249	0.172	0.261	0.169	0.253	0.165	0.260
	336	**0.172**	0.268	0.175	**0.262**	0.197	0.286	0.193	0.281	**0.172**	0.269	0.190	0.279	0.185	0.268	0.176	0.271
	720	**0.204**	0.295	**0.204**	**0.291**	0.236	0.320	0.232	0.329	0.209	0.299	0.231	0.313	0.221	0.300	**0.204**	0.296
Traffic	96	**0.389**	0.260	0.407	**0.252**	0.423	0.286	0.443	0.298	0.426	0.292	0.471	0.295	0.475	0.279	0.450	0.295
	192	**0.403**	0.265	0.431	**0.262**	0.442	0.295	0.496	0.323	0.439	0.298	0.480	0.300	0.486	0.277	0.489	0.311
	336	**0.414**	**0.269**	0.456	**0.269**	0.473	0.335	0.520	0.330	0.449	0.307	0.496	0.306	0.500	0.280	0.484	0.321
	720	**0.445**	**0.285**	0.509	0.292	0.481	0.357	0.540	0.344	0.475	0.309	0.531	0.328	0.537	0.295	0.517	0.333

Table 8: Full TFB results, model group 2. Lower is better; red bold and blue underlining indicate the best and second-best distinct values within this table, including all ties at the displayed precision.

Dataset	H	Aurora-X		Fredformer		MSGNet		iTransformer		PatchTST		Crossformer		TimesNet		DLinear	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.362**	**0.390**	0.378	0.395	0.390	0.411	0.386	0.405	0.414	0.419	0.423	0.448	0.384	0.402	0.397	0.412
	192	**0.392**	**0.409**	0.435	0.424	0.443	0.442	0.441	0.436	0.460	0.445	0.471	0.474	0.436	0.429	0.446	0.441
	336	**0.404**	**0.416**	0.485	0.447	0.482	0.469	0.487	0.458	0.501	0.466	0.570	0.546	0.491	0.469	0.489	0.467
	720	**0.401**	**0.427**	0.496	0.472	0.496	0.488	0.503	0.491	0.500	0.488	0.653	0.621	0.521	0.500	0.513	0.510
ETTh2	96	**0.291**	**0.334**	**0.291**	0.342	0.329	0.371	0.297	0.349	0.302	0.348	0.745	0.584	0.340	0.374	0.340	0.394
	192	**0.349**	**0.373**	0.372	0.390	0.402	0.414	0.380	0.400	0.388	0.400	0.877	0.656	0.402	0.414	0.482	0.479
	336	**0.375**	**0.394**	0.419	0.431	0.440	0.445	0.428	0.432	0.426	0.433	1.043	0.731	0.452	0.452	0.591	0.541
	720	**0.396**	**0.420**	0.431	0.450	0.480	0.477	0.427	0.445	0.431	0.446	1.104	0.763	0.462	0.468	0.839	0.661
ETTm1	96	**0.285**	**0.332**	0.326	0.361	0.319	0.366	0.334	0.368	0.329	0.367	0.404	0.426	0.338	0.375	0.346	0.374
	192	**0.321**	**0.354**	0.363	0.384	0.377	0.397	0.377	0.391	0.367	0.385	0.450	0.451	0.374	0.387	0.382	0.391
	336	**0.347**	**0.371**	0.395	0.406	0.417	0.422	0.426	0.420	0.399	0.410	0.532	0.515	0.410	0.411	0.410	0.415
	720	**0.387**	**0.398**	0.456	0.441	0.487	0.463	0.491	0.459	0.454	0.439	0.666	0.589	0.478	0.450	0.473	0.451
ETTm2	96	**0.174**	**0.248**	0.177	0.258	0.182	0.266	0.180	0.264	0.175	0.259	0.287	0.366	0.187	0.267	0.193	0.293
	192	**0.235**	**0.291**	0.243	0.301	0.248	0.306	0.250	0.309	0.241	0.302	0.414	0.492	0.249	0.309	0.284	0.361
	336	**0.290**	**0.329**	0.302	0.340	0.312	0.346	0.311	0.348	0.305	0.343	0.597	0.542	0.321	0.351	0.382	0.429
	720	**0.371**	**0.384**	0.404	0.398	0.414	0.404	0.412	0.407	0.402	0.400	1.730	1.042	0.408	0.403	0.558	0.525
Weather	96	**0.148**	**0.189**	0.163	0.207	0.163	0.212	0.174	0.214	0.177	0.218	0.158	0.230	0.172	0.220	0.195	0.252
	192	**0.185**	**0.226**	0.224	0.258	0.211	0.254	0.221	0.254	0.225	0.259	0.206	0.277	0.219	0.261	0.237	0.295
	336	**0.226**	**0.260**	0.278	0.298	0.273	0.299	0.278	0.296	0.278	0.297	0.272	0.335	0.280	0.306	0.282	0.331
	720	**0.286**	**0.304**	0.357	0.350	0.351	0.348	0.358	0.349	0.354	0.348	0.398	0.418	0.365	0.359	0.345	0.382
Solar	96	0.201	**0.225**	**0.189**	0.236	0.210	0.246	0.203	0.237	0.234	0.286	0.310	0.331	0.250	0.292	0.290	0.378
	192	**0.223**	**0.241**	0.227	0.259	0.265	0.290	0.233	0.261	0.267	0.310	0.734	0.725	0.296	0.318	0.320	0.398
	336	**0.238**	**0.252**	0.243	0.286	0.294	0.318	0.248	0.273	0.290	0.315	0.750	0.735	0.319	0.330	0.353	0.415
	720	**0.225**	**0.249**	0.250	0.285	0.285	0.315	0.249	0.275	0.289	0.317	0.769	0.765	0.338	0.337	0.357	0.413
Electricity	96	**0.146**	0.244	0.148	0.242	0.165	0.274	0.148	**0.240**	0.195	0.285	0.219	0.314	0.168	0.272	0.210	0.302
	192	**0.159**	0.255	0.165	0.257	0.185	0.292	0.162	**0.253**	0.199	0.289	0.231	0.322	0.184	0.289	0.210	0.305
	336	**0.172**	**0.268**	0.180	0.274	0.197	0.304	0.178	0.269	0.215	0.305	0.246	0.337	0.198	0.300	0.223	0.319
	720	**0.204**	**0.295**	0.218	0.305	0.231	0.332	0.225	0.317	0.256	0.337	0.280	0.363	0.220	0.320	0.258	0.350
Traffic	96	**0.389**	**0.260**	0.403	0.274	0.608	0.349	0.395	0.268	0.544	0.359	0.522	0.290	0.593	0.321	0.650	0.396
	192	**0.403**	**0.265**	0.429	0.289	0.634	0.371	0.417	0.276	0.540	0.354	0.530	0.293	0.617	0.336	0.598	0.370
	336	**0.414**	**0.269**	0.441	0.295	0.669	0.388	0.433	0.283	0.551	0.358	0.558	0.305	0.629	0.336	0.605	0.373
	720	**0.445**	**0.285**	0.463	0.300	0.729	0.420	0.467	0.302	0.586	0.375	0.589	0.328	0.640	0.350	0.645	0.394

A.10 Complete Covariate Forecasting Results

Tables 9 and 10 report both forecast horizons for each DAG-Bench dataset. Table 6 summarizes DAG, GCGNet, TimeXer, TFT, DUET, Amplifier, and TimeKAN. Here, all ten supervised baselines, including TiDE, CrossLinear, and PatchTST, are divided into two groups with Aurora-X as the first model in each. Each table ranks its displayed models separately, including all ties at the reported precision. Aurora-X achieves the lowest horizon-averaged MSE and MAE on all four SDWPF subsets.

Table 9: Full DAG-Bench results, model group 1. Lower is better; red bold and blue underlining indicate the best and second-best distinct values within this table, including all ties at the displayed precision.

Dataset	H	Aurora-X		DAG		GCGNet		TimeXer		TFT		TiDE	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
NP	24	0.231	0.258	**0.202**	**0.237**	0.208	**0.237**	0.236	0.266	0.219	0.249	0.284	0.301
	360	**0.487**	**0.434**	0.521	0.451	0.531	0.459	0.600	0.475	0.539	0.501	0.601	0.498
PJM	24	0.073	0.172	**0.057**	**0.143**	0.060	0.150	0.075	0.166	0.095	0.195	0.106	0.214
	360	0.150	0.236	0.130	**0.218**	**0.129**	0.223	0.140	0.231	0.133	0.219	0.177	0.279
BE	24	**0.348**	0.234	0.361	**0.229**	0.350	0.248	0.392	0.253	0.426	0.272	0.426	0.285
	360	**0.477**	0.331	0.485	0.330	0.511	0.340	0.512	0.327	0.482	**0.310**	0.571	0.364
FR	24	0.358	0.187	0.355	**0.171**	**0.347**	0.188	0.366	0.208	0.543	0.253	0.418	0.255
	360	**0.464**	0.262	0.473	0.268	0.482	0.279	0.489	0.273	0.465	**0.261**	0.551	0.308
DE	24	0.358	0.379	**0.277**	**0.322**	0.280	0.331	0.339	0.362	0.380	0.383	0.367	0.383
	360	0.495	0.439	**0.462**	**0.418**	0.523	0.447	0.610	0.474	0.599	0.509	0.630	0.511
Energy	24	0.137	0.288	**0.079**	**0.215**	0.081	0.221	0.122	0.273	0.093	0.235	0.103	0.248
	360	0.257	0.398	0.169	**0.320**	0.182	0.334	0.204	0.357	**0.167**	0.331	0.202	0.355
Sdwpfm1	24	**0.321**	**0.373**	0.351	0.400	0.376	0.415	0.558	0.533	0.366	0.421	0.474	0.488
	360	**0.339**	**0.403**	0.495	0.522	0.472	0.499	0.845	0.684	0.597	0.528	0.492	0.526
Sdwpfm2	24	**0.333**	**0.387**	0.372	0.414	0.421	0.441	0.627	0.570	0.411	0.458	0.461	0.492
	360	**0.347**	**0.411**	0.583	0.556	0.529	0.531	0.978	0.736	0.541	0.519	0.511	0.540
Sdwpfh1	24	**0.314**	**0.358**	0.408	0.438	0.435	0.486	0.651	0.587	0.401	0.460	0.434	0.489
	360	**0.310**	**0.373**	0.489	0.534	0.465	0.514	0.841	0.700	0.557	0.523	0.472	0.527
Sdwpfh2	24	**0.311**	**0.374**	0.438	0.465	0.473	0.506	0.820	0.677	0.474	0.493	0.579	0.553
	360	**0.312**	**0.391**	0.608	0.595	0.566	0.565	0.962	0.761	0.657	0.549	0.619	0.614
Colbun	10	**0.059**	0.112	0.061	**0.094**	0.065	0.108	0.113	0.172	0.092	0.135	0.089	0.131
	30	**0.126**	**0.178**	0.135	0.215	0.149	0.243	0.176	0.299	0.383	0.460	0.240	0.322
Rapel	10	0.165	0.213	**0.151**	**0.203**	0.211	0.230	0.301	0.308	0.201	0.253	0.228	0.271
	30	**0.286**	**0.345**	0.309	0.408	0.401	0.384	0.387	0.416	0.409	0.414	0.411	0.432

Table 10: Full DAG-Bench results, model group 2. Lower is better; red bold and blue underlining indicate the best and second-best distinct values within this table, including all ties at the displayed precision.

Dataset	H	Aurora-X		DUET		CrossLinear		Amplifier		TimeKAN		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
NP	24	0.231	**0.258**	0.246	0.287	**0.210**	0.266	0.252	0.303	0.273	0.310	0.249	0.294
	360	**0.487**	**0.434**	0.576	0.528	0.531	0.508	0.587	0.534	0.538	0.529	0.530	0.498
PJM	24	0.073	0.172	**0.072**	**0.166**	0.088	0.191	0.096	0.208	0.115	0.244	0.116	0.239
	360	0.150	0.236	**0.131**	**0.228**	0.135	0.254	0.177	0.285	0.162	0.281	0.149	0.279
BE	24	**0.348**	**0.234**	0.432	0.272	0.391	0.259	0.471	0.339	0.451	0.319	0.452	0.326
	360	**0.477**	**0.331**	0.597	0.436	0.568	0.416	0.646	0.487	0.645	0.495	0.702	0.538
FR	24	**0.358**	**0.187**	0.384	0.251	0.390	0.226	0.459	0.348	0.454	0.296	0.518	0.368
	360	**0.464**	**0.262**	0.607	0.403	0.575	0.370	0.648	0.468	0.641	0.452	0.658	0.452
DE	24	**0.358**	0.379	0.376	**0.378**	0.387	0.396	0.394	0.407	0.399	0.412	0.413	0.412
	360	**0.495**	**0.439**	0.589	0.482	0.583	0.507	0.551	0.474	0.547	0.479	0.589	0.498
Energy	24	0.137	0.288	**0.117**	**0.283**	0.241	0.418	0.138	0.306	0.135	0.298	0.239	0.390
	360	0.257	0.398	0.288	0.452	0.237	0.385	0.328	0.472	0.302	0.464	**0.214**	**0.363**
Sdwpfm1	24	**0.321**	**0.373**	0.551	0.564	0.355	0.473	0.364	0.445	0.418	0.503	0.374	0.454
	360	**0.339**	**0.403**	0.646	0.577	0.497	0.532	0.510	0.535	0.476	0.566	0.497	0.550
Sdwpfm2	24	**0.333**	**0.387**	0.445	0.452	0.477	0.536	0.394	0.462	0.474	0.538	0.418	0.492
	360	**0.347**	**0.411**	0.584	0.528	0.589	0.611	0.587	0.563	0.520	0.589	0.602	0.603
Sdwpfh1	24	**0.314**	**0.358**	0.527	0.513	0.548	0.585	0.576	0.627	0.511	0.582	0.422	0.505
	360	**0.310**	**0.373**	0.551	0.519	0.566	0.601	0.497	0.569	0.643	0.694	0.513	0.548
Sdwpfh2	24	**0.311**	**0.374**	0.629	0.563	0.468	0.540	0.473	0.533	0.580	0.614	0.457	0.527
	360	**0.312**	**0.391**	0.665	0.569	0.608	0.609	0.569	0.628	0.713	0.729	0.641	0.632
Colbun	10	**0.059**	0.112	0.089	0.134	0.071	0.102	0.071	0.121	0.061	**0.101**	0.062	0.122
	30	**0.126**	**0.178**	0.307	0.397	0.182	0.288	0.275	0.370	0.195	0.249	0.417	0.496
Rapel	10	0.165	0.213	0.174	0.219	**0.163**	**0.209**	0.181	0.227	0.174	0.231	**0.163**	0.218
	30	**0.286**	**0.345**	0.365	0.432	0.340	0.417	0.333	0.416	0.325	0.390	0.374	0.445

A.11 Token Budget and Effective Context Length

Resolution-adaptive inference offers two ways to allocate a context budget: compress a given history into fewer tokens, or expose the model to a longer history without increasing the number of context tokens. We consider both settings on the eight TFB datasets, keeping the model checkpoint and forecast horizons fixed. Figure 14 illustrates the two operating modes.

Evaluation design. We vary the temporal span per token, s, over $\{48, 96, 192, 336, 480\}$. Each token represents s original time points resampled to the native patch length of 48. Forecasts are restored to the original grid before scoring. MSE and MAE are averaged over the four horizons within each dataset, then equally across the eight datasets. All configurations share forecast origins with sufficient observed history, input settings, and quantile queries. Token counts are per variable.

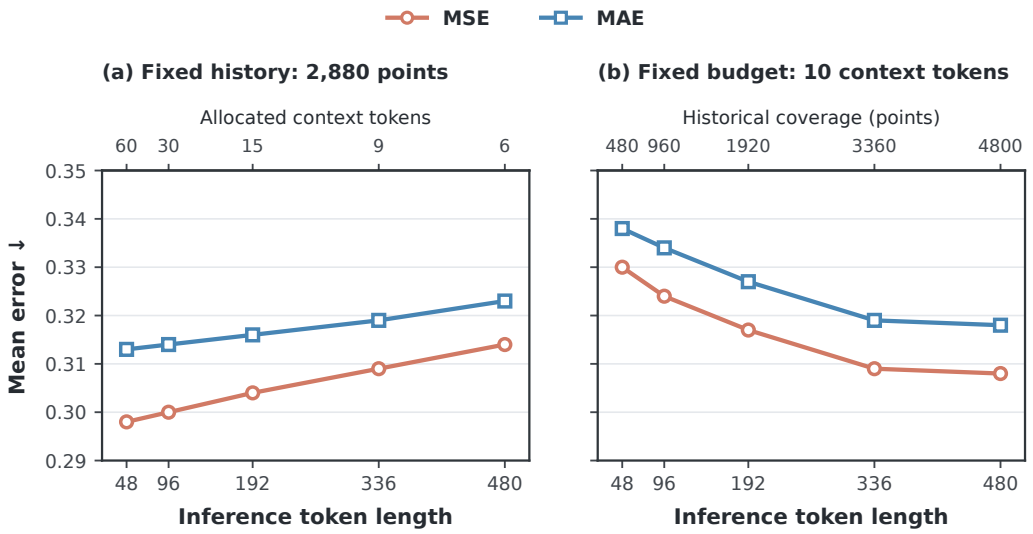


Figure 14: TFB token-budget study. (a) A fixed 2,880-point input with decreasing context-token usage. (b) Ten context tokens with increasing historical coverage. Scores represent averages over eight datasets and four forecast horizons; lower is better.

Compressing a fixed history. With 2,880 supplied observations, the five spans allocate 60, 30, 15, 9, and 6 context-token slots, respectively. Increasing s maps more original points to each native 48-point patch, lowering temporal resolution while preserving embedding dimensions and model weights. A history of T points therefore uses $N_s = \lceil T/s \rceil$ context slots. Fewer tokens reduce temporal-attention interactions and token-wise projection, routing, and expert computation. This sweep tests whether coarser representations retain predictive information as context-token usage decreases by up to 90%, using native resolution as the accuracy reference. The same span applies to every dataset, including Solar. At $s = 336$, the incomplete leading patch is masked: eight valid patches encode the latest 2,688 points, while normalization uses all 2,880 supplied observations.

Extending history at a fixed context budget. With ten context tokens, increasing the span exposes the model to 480, 960, 1,920, 3,360, and 4,800 historical points. This sweep examines whether broader temporal coverage supplies predictive information absent from shorter histories. Together, the two sweeps test the trade-off between temporal detail and historical coverage enabled by variable-resolution post-training. The controlled budget concerns context tokens: for a forecast horizon H, the model still allocates $\lceil H/s \rceil$ future tokens. End-to-end compute therefore also depends on the forecast horizon and resampling operations.

A.12 INFERENCE EFFICIENCY

Experimental setup. We evaluate the cost of a complete probabilistic forecast, including the backbone and its native output procedure. The primary comparison uses the same univariate ETT history with context length $L = 960$, forecast horizon $H = 96$, batch size one, and nine requested quantiles $\{0.1, 0.2, \ldots, 0.9\}$. Here, L counts original observations, so equal history lengths need not imply equal token counts. All models run in FP32 on one NVIDIA A800 80GB PCIe GPU, with TF32, autocast, and `torch.compile` disabled. Table 11 reports six models, including three inference token lengths for Aurora-X.

Table 11: Inference cost at $L = 960$, $H = 96$, batch size one, and nine requested quantiles, measured in FP32 on an A800 80GB PCIe GPU. Parentheses give Aurora-X's inference token length. Parameters are total counts; memory is peak allocated GPU memory. Moirai-MoE uses 100 Monte Carlo paths, and TimesFM uses flip averaging.

Model	Params (B)	GMACs	Latency (ms)	Memory (GiB)
Aurora-X (48)	1.049	8.031	47.054	3.917
Aurora-X (96)	1.049	4.070	46.301	3.917
Aurora-X (192)	1.049	2.329	44.704	3.917
Toto-2.0-1B	1.041	34.114	110.639	4.110
Toto-2.0-2.5B	2.454	80.596	140.409	9.453
Timer-S1	8.304	32.844	107.580	31.268
Moirai-MoE-Base	0.935	2883.643	829.886	3.765
TimesFM-2.5	0.231	13.963	68.689	0.881

Native forecasting procedures. Aurora-X directly queries the nine IQN levels, while Toto uses its official single-pass parallel forecasting configuration. Timer-S1 retains native generation with RevIN, KV caching, and the executed multi-token prediction layers. Moirai-MoE uses patches of length 16 and 100 Monte Carlo paths to obtain empirical quantiles; at $H = 96$, its initial forward pass is followed by five full forward passes over these paths. TimesFM retains its continuous quantile head, flip averaging with two decoding passes, and native output corrections. Only its wrapper-level CPU conversions are removed, with predictions checked against the original API. These settings preserve each model's forecasting procedure; the common quantile request does not equate their predictive distributions or accuracy.

Measurement protocol. Matrix multiply–accumulate operations (MACs) are counted from executed linear layers, attention, and routed experts using `FlopCounterMode` with an independent operator audit. The matrix multiplication within Moirai's distance-based routing is included. Attention counts use complete dense matrix shapes, including masked positions. One GMAC denotes 10^9 MACs, with two FLOPs per matrix MAC. Normalization, activation, sampling, sorting, and interpolation contribute to latency but are excluded from matrix MACs. Counting uses `no_grad`, and its outputs are verified against native inference.

Latency is the median synchronized wall-clock time under `inference_mode`, following five warm-up iterations. We use 30 repetitions for Aurora-X's token-length sweep and 20 for each baseline. Inputs are already on the GPU; timing includes normalization, routing, prediction heads, and complete native generation, excluding weight loading and host–device transfers. The measured GPU has no competing compute process. Memory is PyTorch's peak allocated GPU memory, including model parameters and tensors. Parameter counts include all loaded parameters, excluding buffers, rather than only the active MoE parameters.

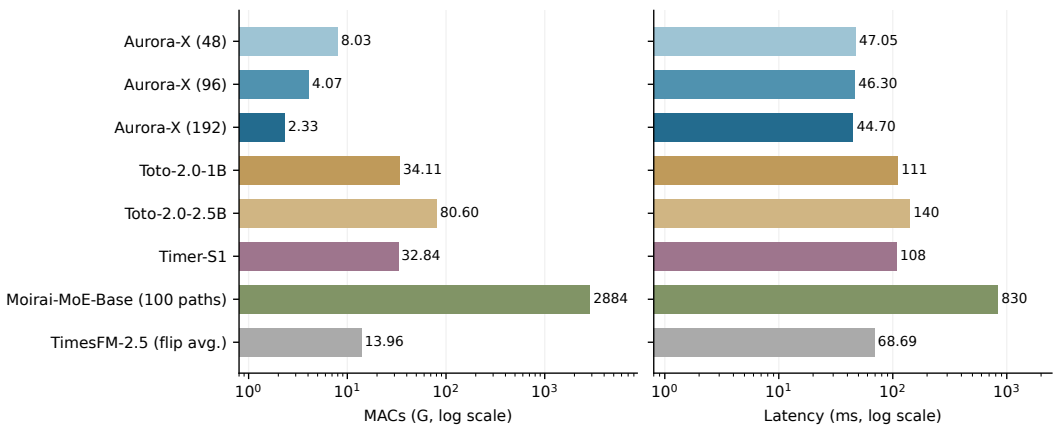


Figure 15: Complete-forecast MACs and median latency for the configurations in Table 11. Native output procedures are retained, including Moirai-MoE's 100 sampling paths and TimesFM's flip averaging.

Comparison with pretrained forecasters. Figure 15 compares the resulting computation and latency. With comparable total parameter counts, Aurora-X at its native token length of 48 uses 76.5% fewer MACs than Toto-2.0-1B and achieves a measured 2.35× speedup. Timer-S1 has a larger total parameter count, whereas TimesFM provides a smaller-model reference. Moirai-MoE's repeated sampling contributes substantially to its complete-forecast cost, so this difference cannot be attributed solely to the backbone architecture.

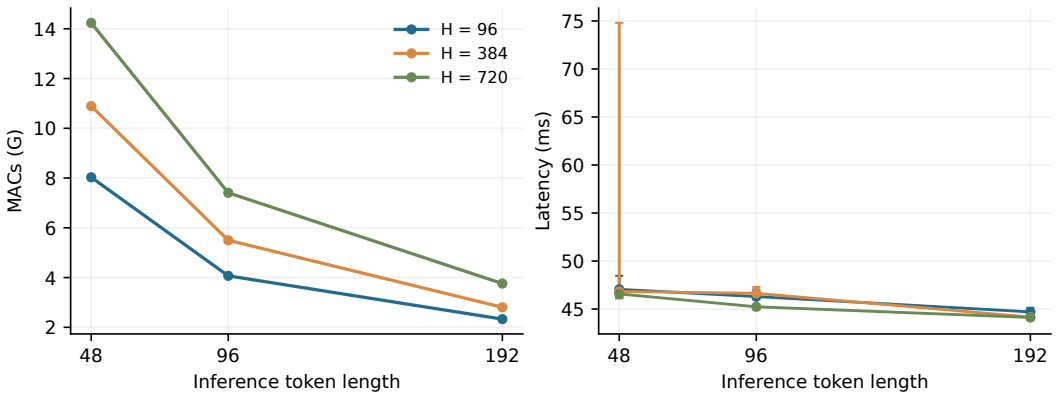


Figure 16: Aurora-X inference cost as the temporal span per token increases, with $L = 960$ and three forecast horizons. Each configuration returns nine quantiles. Latency error bars show the empirical 10th–90th percentiles of 30 synchronized runs.

Effect of temporal resolution. Figure 16 complements the context-budget study in Appendix A.11 with measured computation and latency. Longer inference tokens lower temporal resolution by resampling longer blocks to the native patch length of 48. At $L = 960$ and $H = 96$, lengths 48, 96, and 192 use 20, 10, and 5 context tokens, plus 2, 1, and 1 future tokens, respectively. This shorter backbone sequence reduces attention, routing, and expert computation; output resampling still returns all 96 forecast points. Lengths 96 and 192 reduce MACs by 49.3% and 71.0%, respectively, relative to length 48. Native output padding and final truncation remain included in the measurement. The 3.45× reduction in MACs at length 192 yields a much smaller 1.05× latency speedup, demonstrating that matrix computation alone does not determine end-to-end latency. This efficiency sweep does not evaluate the corresponding change in benchmark accuracy.

A.13 Forecasting with Arbitrary Quantile Queries

The implicit quantile head allows users to query forecast levels beyond a predefined output grid. Figures 17 and 18 illustrate this interface on eight real forecasting windows from GIFT-Eval, TIME, FEV-Bench, and TFB. Each example combines five directly queried random quantile levels with an independently computed central forecast summary. Appendix A.14 complements these examples with point and interval forecasts on all predefined windows.

Case selection. We use 24 predefined test windows from twelve dataset configurations. For each window, we score the predetermined display variable by the mean absolute error of its central forecast, divided by the population standard deviation of its cached input history. We retain the lowest-error window per dataset and then the two lowest-error datasets per benchmark; ties use normalized RMSE and the case identifier. This selection uses test targets, yielding illustrative examples with relatively accurate central forecasts rather than an unbiased performance summary. Table 12 lists the resulting inputs.

Table 12: Input configurations of the eight selected forecasting examples. L is cached input length, H is the forecast horizon, and C is the number of jointly predicted target variables. Each plot displays the listed variable and the last $2H$ history points.

Panel	Benchmark	Dataset	Variable	L	H	C
(a)	GIFT-Eval	jena_weather/H	var_0	8112	48	21
(b)		solar/H	var_0	7992	48	1
(c)	TIME	Australia_Solar/H	solar_63726	2880	24	3
(d)		epf_electricity_price/H	var_0	2880	24	1
(e)	FEV-Bench	bizitobs_l2c_1H	target_0	2280	24	7
(f)		ETT_1H	OT	8160	168	7
(g)	TFB	ETTh1	OT	528	96	7
(h)		Weather	OT	528	96	21

Quantile queries and central forecasts. For each case, five levels are sampled independently from $\mathrm{Uniform}(0.02, 0.98)$, rounded to four decimal places, and queried directly through the IQN head. We use a fixed global seed of 20260921 with deterministic case-specific seeds. Separately, for each forecast step h, we average predictions at twenty equally spaced levels in $[0.05, 0.95]$:

$$\bar{y}_h^{(20)} = \frac{1}{20} \sum_{j=0}^{19} \widehat{Q}_h \left(0.05 + \frac{0.90j}{19} \right). \tag{14}$$

Here $\widehat{Q}_h(\tau)$ denotes the predicted marginal quantile at level τ. This finite-grid average is neither the median nor an exact integral over the full predictive distribution. The five random levels do not enter its calculation. Requested levels are sorted for display, while the predicted values and any quantile crossings are preserved.

Inference and interpretation. All eight cases use the same Aurora-X checkpoint in FP32, with joint target-variable attention and an inference token length of 48. Model inputs retain the cached histories in Table 12; incomplete leading patches follow native masking. TFB inputs are standardized using training-split statistics, and predictions are restored to their original units. Plots show only the last $2H$ observed steps before the forecast origin. Solar examples exhibit larger separation between quantile curves near production peaks, whereas the displayed bizitobs window has a narrower spread. These views illustrate how directly queried marginal forecasts vary with the input and forecast step. The curves are not joint trajectory samples, and the selected examples do not establish distributional calibration.

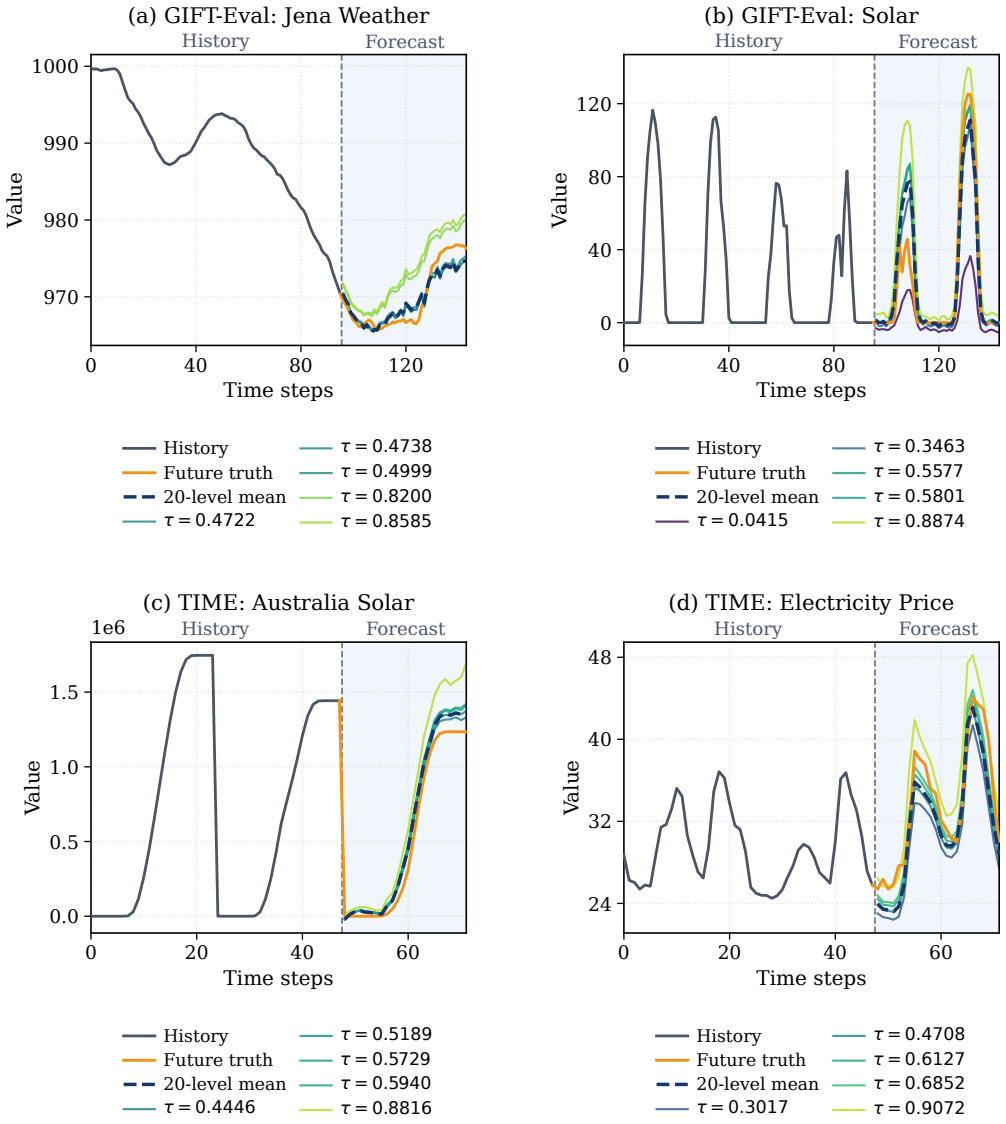


Figure 17: Selected forecasting examples on GIFT-Eval and TIME. Gray and orange denote observed history and future truth. Thin colored curves show five random marginal quantiles; dashed blue shows the independent twenty-level average. The vertical dashed line separates history from forecast. Cases are selected by test error as described in Appendix A.13.

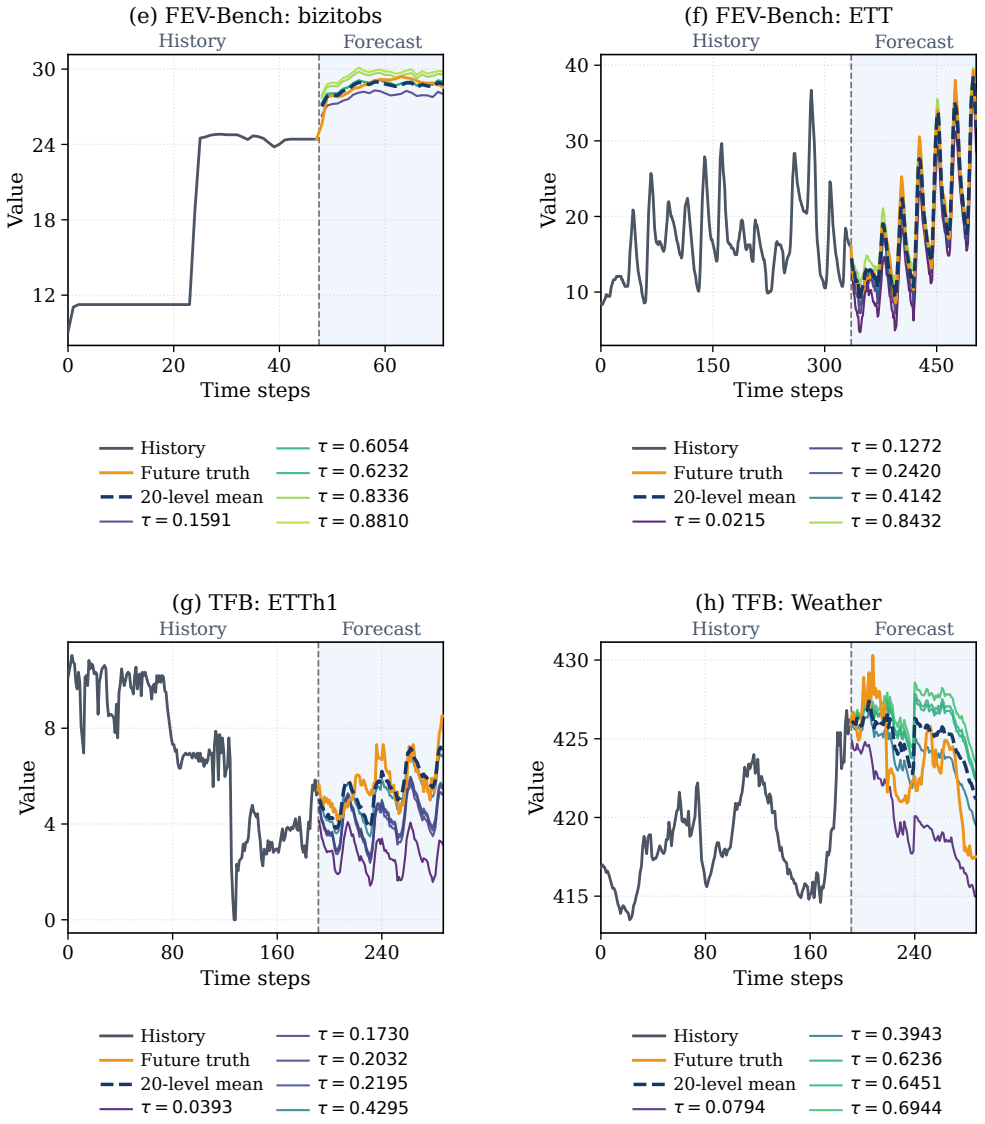


Figure 18: Selected forecasting examples on FEV-Bench and TFB, using the same display and test-error selection protocol as Figure 17. Each panel preserves $2H$ history points, the full H-step forecast, and the five originally queried quantile levels. Shading identifies the forecast window.

A.14 Additional Forecasting Visualizations

Figures 19–22 show all 24 predefined test windows from twelve dataset configurations, with two windows per configuration. Windows and display variables were fixed before inference, without filtering by forecast error. Each panel compares Aurora-X with Seasonal Naive, which repeats the latest observed seasonal cycle. The blue forecast averages nineteen queried levels, $0.05, 0.10, \ldots, 0.95$; the shaded band spans P10–P90 and denotes a nominal 80% prediction interval. Only these two endpoints are reordered when necessary for plotting. Inputs retain all cached history and jointly modeled targets; displayed values are in original units. The inference token length is 48, except for the two short-history Australian Tourism cases, which use 8. These views complement the arbitrary-quantile examples with recurring cycles, level changes, and irregular fluctuations; empirical interval coverage varies across windows.

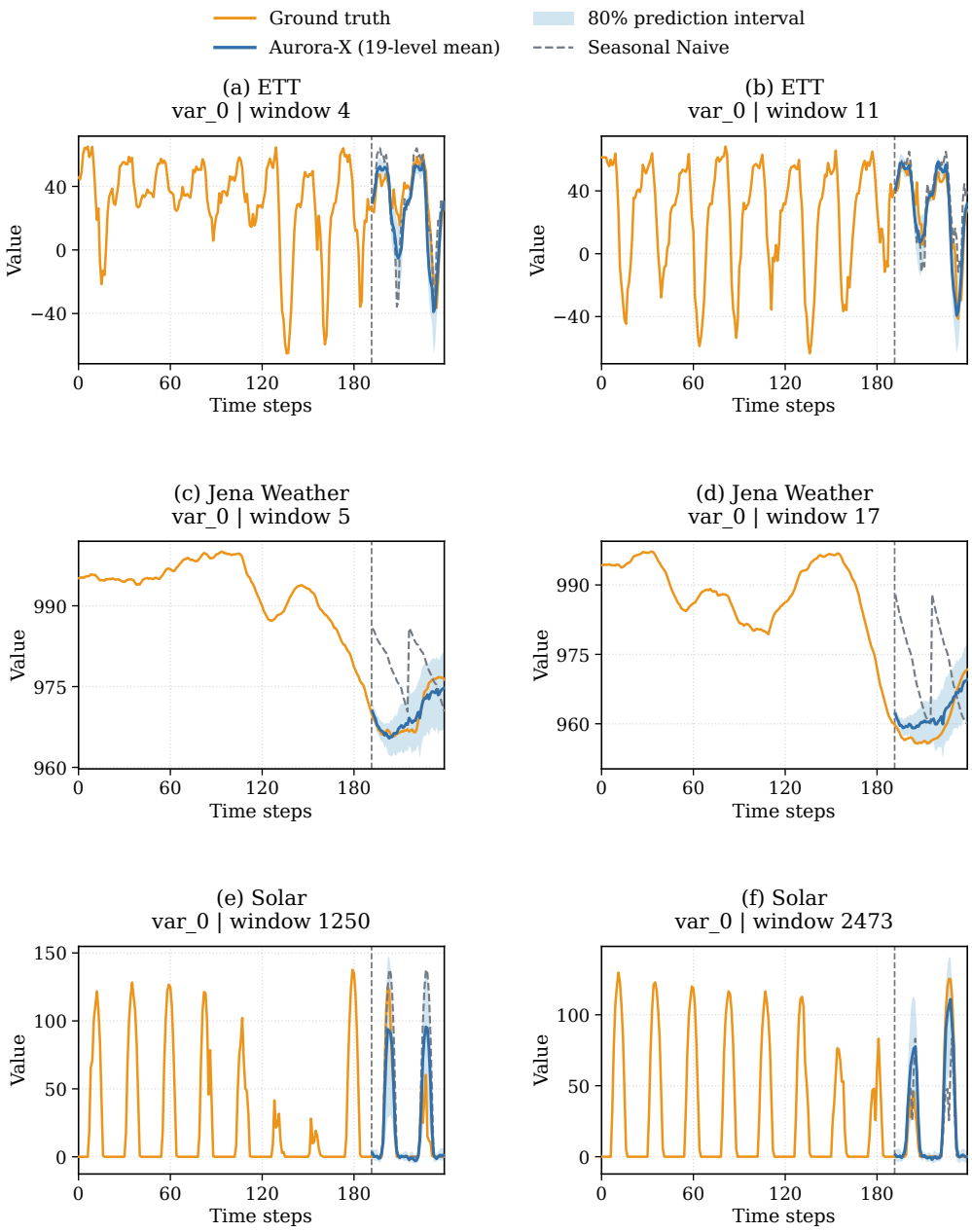


Figure 19: Forecasts on six predefined GIFT-Eval windows from ETT, Jena Weather, and Solar, with horizon 48. Orange shows observed history and future truth; blue is Aurora-X's nineteen-level mean; gray dashed curves show Seasonal Naive. Shading denotes the P10–P90 prediction interval. The vertical dashed line marks the forecast origin. Each panel displays 192 history points from the cached input.

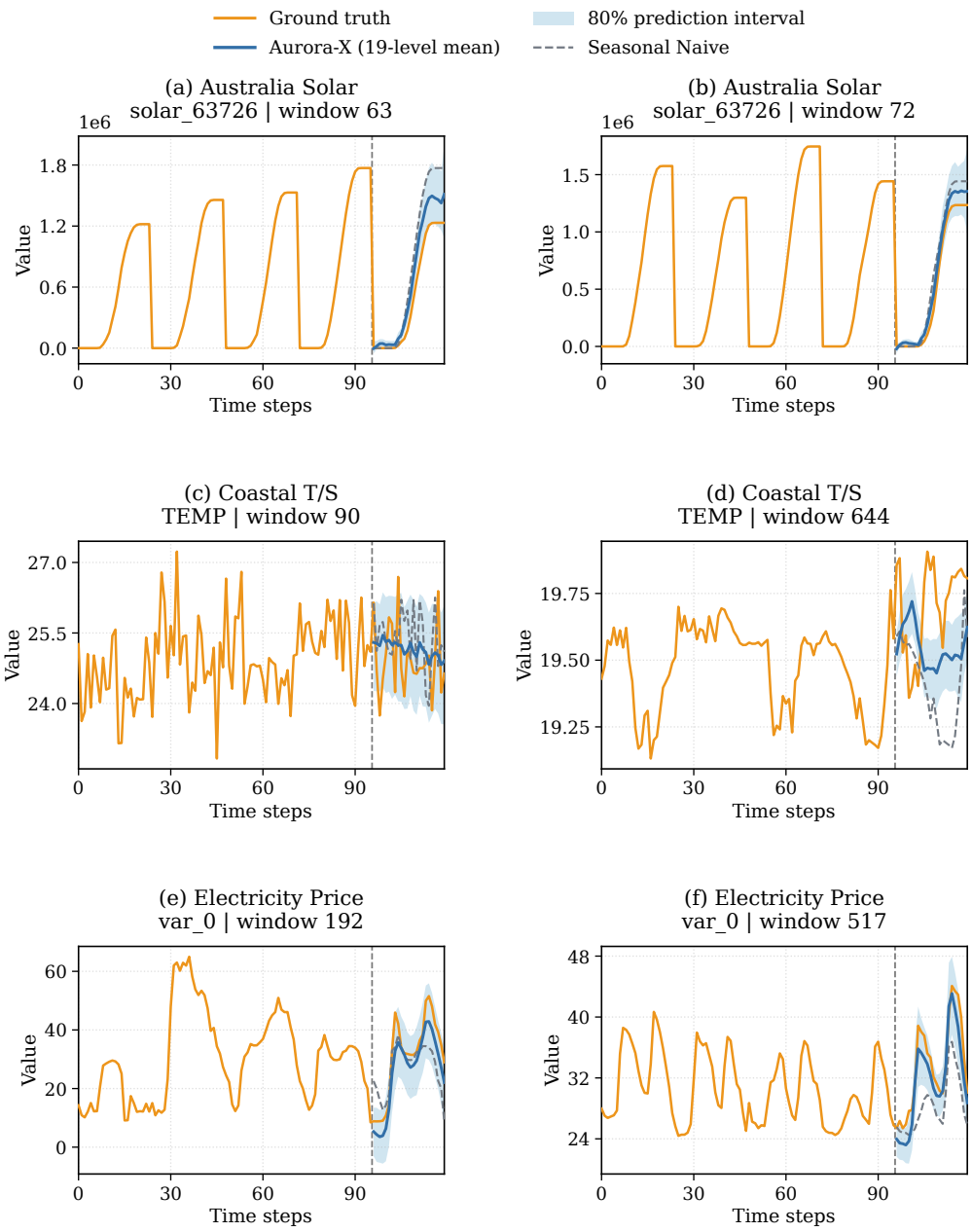


Figure 20: Forecasts on six predefined TIME windows from Australia Solar, Coastal T/S, and electricity prices. Models receive 2,880 history points and predict 24 steps; each panel displays the final 96 history points. Curves and intervals follow Figure 19. These windows show both close forecasts and departures from the observed future, particularly for irregular fluctuations.

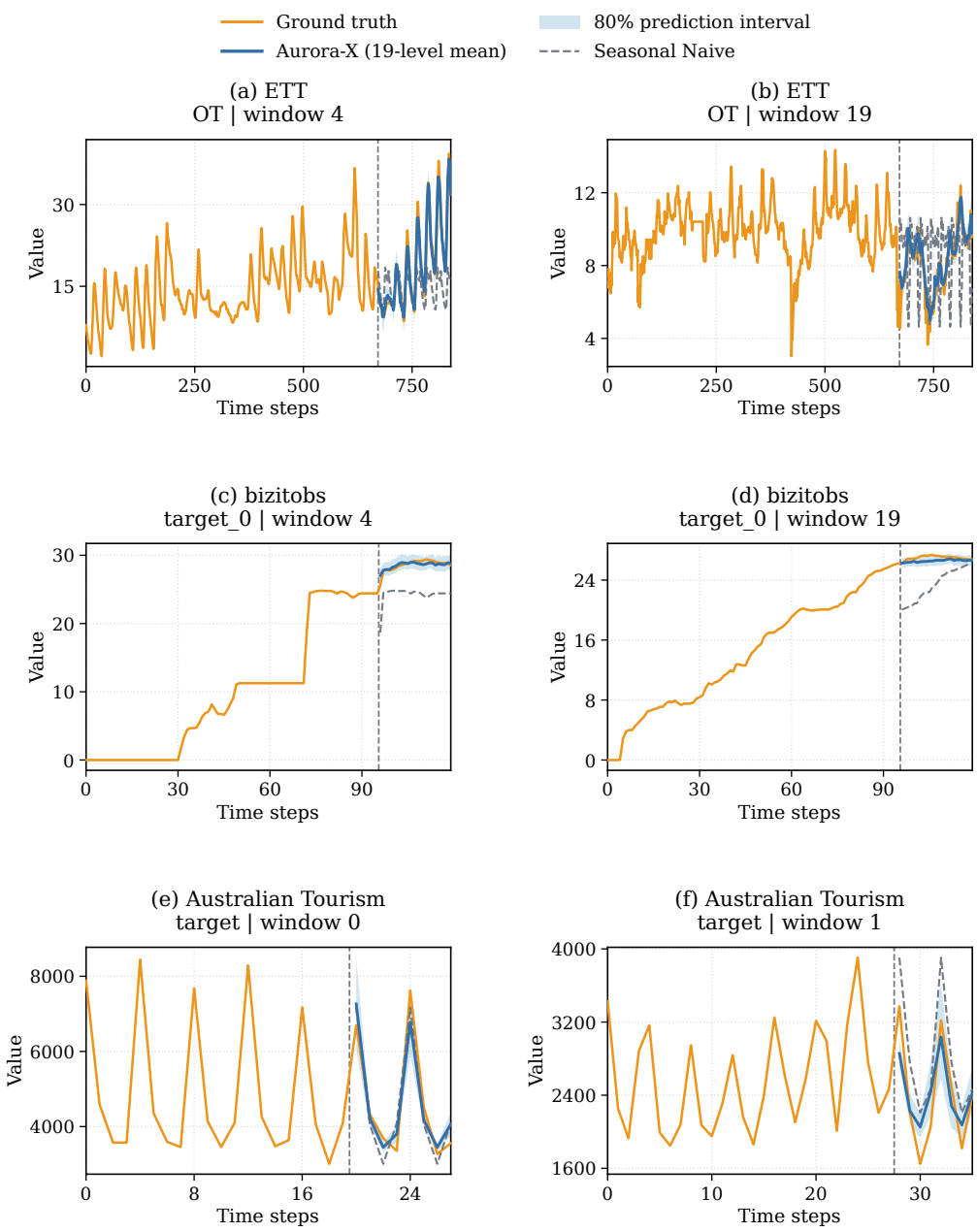


Figure 21: Forecasts on six predefined FEV-Bench windows. Horizons are 168 for ETT, 24 for bizitobs, and 8 for Australian Tourism. The corresponding displays retain 672, 96, and 20 or 28 historical points, respectively. Curves and intervals follow Figure 19; the Tourism cases illustrate prediction from short observed histories.

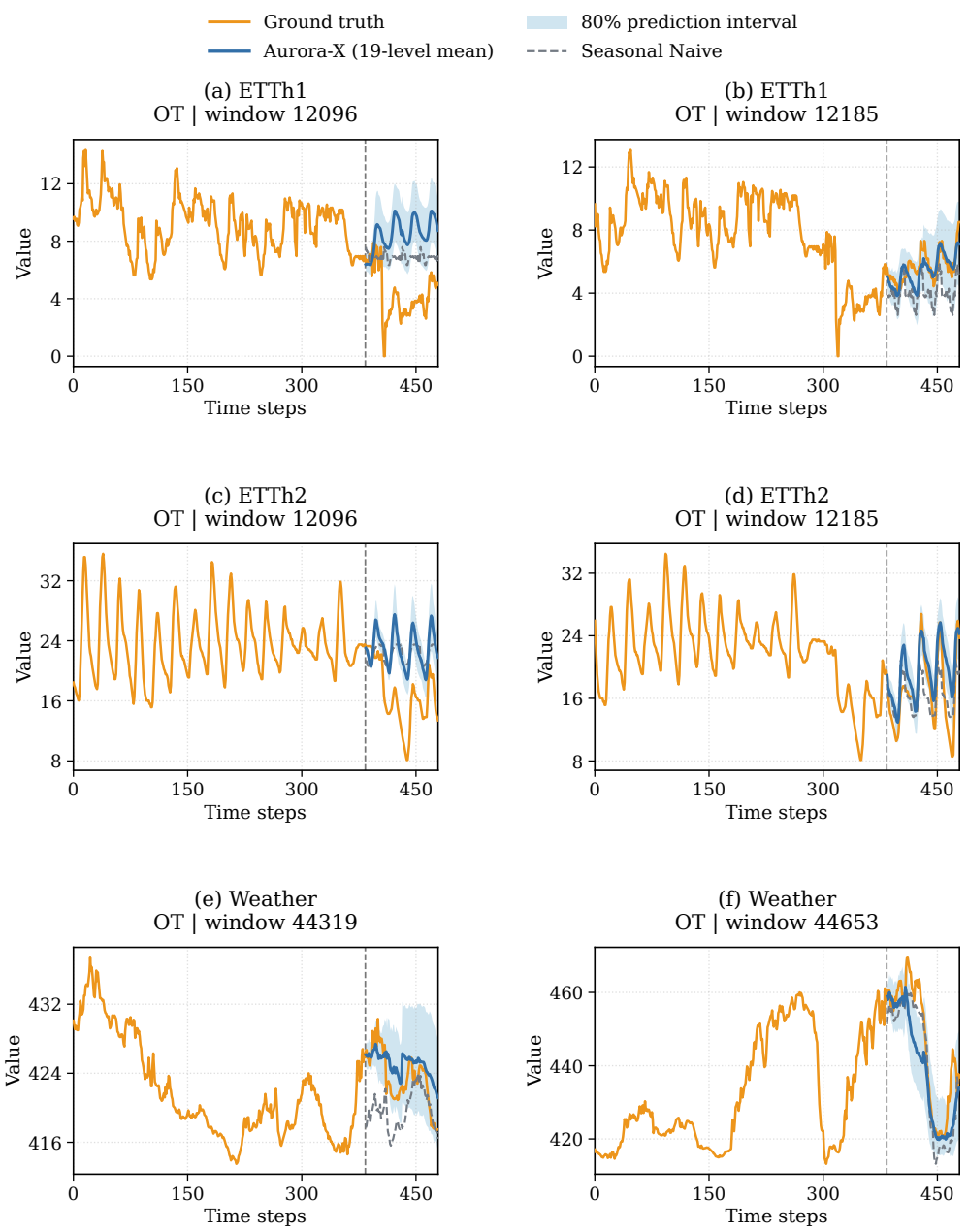


Figure 22: Forecasts on six predefined TFB windows from ETTh1, ETTh2, and Weather, with horizon 96. Each model input contains 528 history points, of which the final 384 are shown. Predictions are restored from training-split standardization to original units. Curves and intervals follow Figure 19.

B Resolution Adaptation and Training Consistency

Resolution adaptation changes the temporal span represented by each token while retaining a pre-trained backbone. This section relates our explicit patch resampling to the projection resizing used in LightGTS (Wang et al., 2025a), then explains how the data-side formulation supports consistent multi-scale training. We compare resolution operators around a shared base model, with matched patch boundaries, normalization, and auxiliary inputs.

B.1 Equivalent Placement of Resolution Operators

Let P be the native patch length and $P_k = kP \in \mathbb{N}$ the requested temporal span. For a normalized input patch $x_k \in \mathbb{R}^{P_k}$ and a native-grid output patch $y \in \mathbb{R}^P$, write linear resampling in matrix form:

$$\begin{aligned} D_k x_k &= \mathcal{R}_{P_k \to P}(x_k), \quad D_k \in \mathbb{R}^{P \times P_k}, \\ U_k y &= \mathcal{R}_{P \to P_k}(y), \quad U_k \in \mathbb{R}^{P_k \times P}. \end{aligned} \tag{15}$$

The matrices use fixed interpolation coordinates and boundary conventions. D_k maps observations to the native grid, and U_k restores predictions to the requested grid; they need not be inverses.

Proposition 1 (data–parameter equivalence). For an input projection $W \in \mathbb{R}^{d \times P}$ with bias b, resampling the patch is exactly equivalent to replacing its value projection by $W_k = WD_k$. Likewise, for an output projection $V \in \mathbb{R}^{P \times d}$ with bias c, output resampling is equivalent to $V_k = U_k V$ and $c_k = U_k c$:

$$\begin{aligned} W(D_k x_k) + b &= (WD_k)x_k + b, \\ U_k(Vh + c) &= (U_k V)h + U_k c. \end{aligned} \tag{16}$$

Proof. Both identities follow from associativity and distributivity of matrix multiplication. In Aurora-X's residual MLP embedding, the same composition is applied to the value columns of both input affine branches; time and mask features are held fixed. Both branches then receive identical pre-activations, so the subsequent nonlinearities, expert routing, and backbone states are identical. On the output side, composing U_k with the quantile head's final affine projection gives the same restored predictions for every quantile query. Thus, the equivalence extends through nonlinear modules when the linear resolution maps are placed at their boundaries.

B.2 Relationship to LightGTS Flex-Resize

LightGTS derives a particular projection transformation using a Moore–Penrose pseudoinverse. With matched interpolation conventions, converting its row-vector notation to the column-vector convention above gives

$$R_k^{\text{Flex}} = \delta_k^{-1} U_k^{+}, \qquad W_k^{\text{Flex}} = W R_k^{\text{Flex}}, \qquad \delta_k = \sqrt{P/P_k}. \tag{17}$$

Here δ_k is the scale factor specified by LightGTS. Consequently, Flex-resize itself has an exact data-side realization:

$$W_k^{\text{Flex}} x_k + b = W(R_k^{\text{Flex}} x_k) + b. \tag{18}$$

Equation (18) reveals the resampling implicit in parameter resizing. The vector $U_k^{+} x_k$ is the minimum-norm least-squares reconstruction of P native-grid values whose interpolation by U_k best fits the P_k observations. For $P_k > P$, it reduces the sample count; for $P_k < P$, it reconstructs a denser native grid. Thus, Flex-resize implements an interpolation-based upsampling or downsampling operation on the input, followed by the scale factor δ_k^{-1}, while keeping that operation implicit in the projection weights. Its pseudoinverse-derived reconstruction weights need not equal ordinary interpolation weights.

Proposition 2 (conditions for agreement with Flex-resize). For a fixed value projection W, direct resampling and Flex-resize produce identical outputs of this affine projection for every x_k if and only if $W(D_k - R_k^{\text{Flex}}) = 0$. Agreement for every value projection requires $D_k = R_k^{\text{Flex}}$. Their projection discrepancy satisfies

$$\begin{aligned} z_k^{\text{direct}} - z_k^{\text{Flex}} &= W(D_k - R_k^{\text{Flex}})x_k, \\ \|z_k^{\text{direct}} - z_k^{\text{Flex}}\|_2 &\le \|W\|_2 \, \|D_k - R_k^{\text{Flex}}\|_2 \, \|x_k\|_2. \end{aligned} \tag{19}$$

Proof. Subtracting the two affine projections cancels their common bias and yields the first identity. Equality for all patches is equivalent to the resulting linear map being zero. Requiring equality for all W further forces $D_k = R_k^{\mathrm{Flex}}$. The bound follows from submultiplicativity of the spectral norm.

Aurora-X uses ordinary linear interpolation for D_k, whereas LightGTS uses the scaled pseudoinverse R_k^{Flex}. Reverse interpolation is generally different from a pseudoinverse. Accordingly, the exact equivalence concerns placement of a matched operator; approximation to the particular LightGTS operator depends on the discrepancy in Equation (19).

For a patch generated as $x_k = U_k x_0$, suppressing the common affine bias and explicitly aligning LightGTS's scalar factor gives

$$z_k^{\mathrm{direct}} - \delta_k z_k^{\mathrm{Flex}} = W(D_k U_k - U_k^{+} U_k) x_0. \tag{20}$$

If U_k has full column rank, then $U_k^{+} U_k = I_P$: the remaining difference is the round-trip interpolation error measured by W. It vanishes on signals recovered by $D_k U_k$ and is small when their recovery error is small. Otherwise, $U_k^{+} U_k$ projects the native signal onto the recoverable subspace. The scalar factor remains part of the implemented Flex-resize; Equation (20) explicitly aligns it for this comparison.

B.3 Consistency of Multi-Scale Training and Inference

The data-side formulation makes the same resolution transformation available for constructing training examples. Let $\mathcal{N}_X$ denote RevIN followed by $\mathrm{arcsinh}$, using statistics from the original observed history X. For known future covariates C and targets Y, construct

$$X_k = \mathcal{A}_k(\mathcal{N}_X(X)), \qquad C_k = \mathcal{A}_k(\mathcal{N}_X(C)), \qquad Y_k = \mathcal{A}_k(\mathcal{N}_X(Y)), \tag{21}$$

where statistics are applied variable-wise and $\mathcal{A}_k$ applies D_k separately to each patch. Each augmented history–target pair therefore uses the same temporal granularity, with P_k original points represented by P native-grid values.

Let F_θ denote the native-grid forecaster, including covariate conditioning and a quantile query τ, and let $\mathcal{B}_k$ apply U_k to each output patch. The corresponding training objective and inference computation are

$$\begin{aligned} \mathcal{J}(\theta) &= \mathbb{E}_{(X,C,Y),k,\tau}\left[\ell_\tau(F_\theta(X_k, C_k; \tau), Y_k)\right], \\ \widehat{Y}_k(\tau) &= \mathcal{N}_X^{-1}(\mathcal{B}_k F_\theta(X_k, C_k; \tau)), \end{aligned} \tag{22}$$

where ℓ_τ averages the masked pinball loss over valid target positions, as in Equation (13). Both computations call the same forecaster on the same canonical inputs. Their consistency follows directly from reusing the augmentation operator in inference, with supervision defined on that canonical grid.

This construction fixes the order of preprocessing: history-based normalization and $\mathrm{arcsinh}$ precede resampling, and the augmented arrays are not normalized again. History blocks are aligned to the forecast origin, and future blocks begin at the first forecast step. History statistics and resampling do not depend on future targets; targets are transformed separately using those history statistics. Related variables share scale and block boundaries. Alignment requires identical validity rules and native-grid time features during training and inference. These choices preserve the input interface even when patches contain different amounts of historical information.

Implications for foundation-model training. The operator equivalence allows resolution adaptation to be expressed as paired data augmentation while retaining a fixed model interface. Multiple resolutions can be prepared in the data pipeline and batched using the same patch shape, embedding, quantile head, and loss. Scales can be enumerated without requiring a detected period, and the same transformation aligns histories, covariates, and targets. LightGTS also learns across varying periods through its adaptive projections; our design emphasizes explicit paired augmentation and dedicated variable-resolution post-training. This uniform data construction provides direct training–inference alignment and a reusable augmentation pipeline, making it well suited to scaling foundation-model training over heterogeneous sequences.

C Theoretical Analysis of Shallow-Pattern Clustering

Expert specialization requires a criterion for deciding which patches should share computation. Similarity-based channel clustering (Chen et al., 2024a) motivates using temporal structure to guide parameter sharing. We develop this connection for Aurora-X's normalized, overlapping expert assignments: the pattern loss measures cluster coherence, its gradient favors sharing among sufficiently similar patches, and the projection regularizer encourages distinct feature directions.

C.1 Normalized Co-Assignment and Kernel Clustering

Consider one nonempty context with N patches at one MoE layer, and omit batch and layer indices. Let $D \in \{0,1\}^{N\times E}$ contain the hard top-K assignments, with $\sum_e D_{ie} = K$, $1 \le K \le E$, and $\sigma_{\mathrm{pat}} > 0$. The shared expert is excluded. Define

$$\Omega = \frac{DD^\top}{K}, \qquad Z_\Omega = \sum_{i,j} \Omega_{ij}, \qquad \rho = \frac{\sum_{i,j} \Omega_{ij} S_{ij}}{Z_\Omega}, \qquad \ell_{\mathrm{pat}} = -\rho. \tag{23}$$

This is the per-context term of Equation (7), including self-pairs. Since $\Omega_{ii} = 1$, $Z_\Omega \ge N$, so a denominator floor smaller than one is inactive. For the kernel geometry below, assume that projected patch norms exceed the normalization floor, giving $\|u_i\|_2 = 1$. Their similarity then satisfies

$$S_{ij} = \exp\left(-\frac{\|u_i - u_j\|_2^2}{2\sigma_{\mathrm{pat}}^2}\right) = \langle \phi_i, \phi_j \rangle_{\mathcal{H}}, \qquad \|\phi_i\|_{\mathcal{H}}^2 = 1, \tag{24}$$

where ϕ_i is a feature representation of the Gaussian kernel in a Hilbert space $\mathcal{H}$.

Proposition C.1 (within-expert kernel dispersion). Let $\mathcal{C}_e = \{i : D_{ie} = 1\}$, $n_e = |\mathcal{C}_e|$, and, for each nonempty expert assignment, define its feature mean and variance by

$$\mu_e = \frac{1}{n_e} \sum_{i\in\mathcal{C}_e} \phi_i, \qquad V_e = \frac{1}{n_e} \sum_{i\in\mathcal{C}_e} \|\phi_i - \mu_e\|_{\mathcal{H}}^2.$$

Then the clustering objective admits the exact decomposition

$$1 + \ell_{\mathrm{pat}} = \frac{\sum_{e:n_e>0} n_e^2 V_e}{\sum_e n_e^2}. \tag{25}$$

Proof. Expanding the variance gives $V_e = 1 - \|\mu_e\|_{\mathcal{H}}^2$. Moreover,

$$K Z_\Omega = \sum_e n_e^2, \qquad K \sum_{i,j} \Omega_{ij} S_{ij} = \sum_e \left\| \sum_{i\in\mathcal{C}_e} \phi_i \right\|_{\mathcal{H}}^2 = \sum_{e:n_e>0} n_e^2 \|\mu_e\|_{\mathcal{H}}^2.$$

Substituting these identities into Equation (23) proves the result. Each patch may belong to several $\mathcal{C}_e$, so the identity applies directly to top-K routing.

Thus, the loss measures a weighted average of within-expert pattern dispersion. Experts receive weights proportional to n_e^2, reflecting their numbers of co-assigned ordered pairs. It encourages coherent expert assignments, while allowing overlapping pattern groups. For zero or sub-unit normalized features, use the feature map of the positive-semidefinite kernel $k(u,v) = \exp[(u^\top v - 1)/\sigma_{\mathrm{pat}}^2]$. The same expansion adds the diagonal-deficit term $\sum_e n_e \sum_{i\in\mathcal{C}_e} (1 - S_{ii}) / \sum_e n_e^2$ to Equation (25). The pure variance interpretation holds under the unit-norm condition.

Corollary (control of dissimilar co-assignments). For $0 < \delta \le 2$, let q_δ be the share of total co-assignment weight allocated to pairs with $\|u_i - u_j\|_2 \ge \delta$. Under the unit-norm condition,

$$q_\delta = \frac{\sum_{i,j} \Omega_{ij} \mathbf{1}[\|u_i - u_j\|_2 \ge \delta]}{Z_\Omega} \le \frac{1 + \ell_{\mathrm{pat}}}{1 - \exp[-\delta^2/(2\sigma_{\mathrm{pat}}^2)]}. \tag{26}$$

Indeed, $1 + \ell_{\mathrm{pat}} = \sum_{i,j} \Omega_{ij}(1 - S_{ij})/Z_\Omega$, and each such pair contributes at least $1 - \exp[-\delta^2/(2\sigma_{\mathrm{pat}}^2)]$. The bound quantifies how a loss close to -1 restricts dissimilar pairs that share experts; its weights include self-pairs, as in the implemented objective.

C.2 Pattern-Dependent Learning Signals for Routing

The normalization by Z_Ω makes the routing signal depend on relative pattern similarity. To characterize this signal, hold S fixed and consider the continuous extension of Equation (23).

Proposition C.2 (normalized affinity and straight-through gradients). The derivative with respect to an overlap entry is

$$\frac{\partial \ell_{\mathrm{pat}}}{\partial \Omega_{ij}} = -\frac{S_{ij} - \rho}{Z_\Omega}. \tag{27}$$

For the actual straight-through assignment in Equation (6), let $p_i = \mathrm{Softmax}(s_i)$, where s_i is the normalized router-logit vector. Define the affinity of patch i to expert e's assigned patches, centered at the current mean similarity, as

$$c_{ie} = \sum_j D_{je}(S_{ij} - \rho).$$

The backward derivative with respect to s_{ie} is

$$\frac{\partial \ell_{\mathrm{pat}}}{\partial s_{ie}} = -\frac{2p_{ie}}{KZ_\Omega}\left(c_{ie} - \sum_f p_{if} c_{if}\right). \tag{28}$$

Proof. The quotient rule gives Equation (27). Writing $\Omega = AA^\top/K$, symmetry of S yields

$$\left.\frac{\partial \ell_{\mathrm{pat}}}{\partial A}\right|_{A=D} = -\frac{2}{KZ_\Omega}(S - \rho \mathbf{1}\mathbf{1}^\top)D.$$

The straight-through construction has $\partial A/\partial p = I$ in the backward pass. Applying the softmax Jacobian $\mathrm{diag}(p_i) - p_i p_i^\top$ gives Equation (28).

Equation (27) favors increasing overlap for pairs more similar than the current co-assignment average and decreasing it for less similar pairs. Equation (28) describes how this preference reaches the router: the partial derivative with respect to an expert's normalized logit is negative when its pattern-affinity score exceeds the probability-weighted mean across experts. Gradients subsequently pass through logit normalization and the shared routing parameters. They are the implemented straight-through surrogate gradients, rather than derivatives of the discrete top-K map.

The similarity matrix is computed from shallow patches, so its pattern distinctions do not directly depend on distinctions retained in deep hidden states. In addition, for fixed assignments, $\partial \ell_{\mathrm{pat}}/\partial S_{ij} = -\Omega_{ij}/Z_\Omega$: the learned pattern projection receives a signal to increase similarity among co-assigned patches. These two paths couple expert sharing with learning a local pattern metric.

C.3 When Pattern Grouping Is Preferred to Collapsed Routing

The preceding identities characterize local training signals. The following comparison states when the objective itself favors differentiated assignments over sending every patch to the same experts.

Proposition C.3 (objective gap for pattern-separated routing). Partition the patches into $G \geq 2$ nonempty pattern groups of sizes n_g, and suppose $E \geq GK$. Hold S fixed. Let μ_{in} and μ_{out} be the mean similarities over within-group and between-group ordered pairs, respectively, including self-pairs within groups. Compare two feasible assignments: each group selects its own disjoint set of K experts, or every patch selects the same K experts. Their pattern losses satisfy

$$\ell_{\mathrm{collapse}} - \ell_{\mathrm{group}} = \left(1 - \frac{\sum_g n_g^2}{N^2}\right)(\mu_{\mathrm{in}} - \mu_{\mathrm{out}}). \tag{29}$$

Consequently, pattern-separated routing has strictly lower loss when $\mu_{\mathrm{in}} > \mu_{\mathrm{out}}$.

Proof. With disjoint expert sets, $\Omega_{ij} = 1$ within a group and zero between groups, so $\ell_{\text{group}} = -\mu_{\text{in}}$. Under complete routing collapse, $\Omega_{ij} = 1$ for every pair. Its mean similarity is therefore $[\sum_g n_g^2 \mu_{\text{in}} + (N^2 - \sum_g n_g^2)\mu_{\text{out}}]/N^2$. Subtracting the losses gives Equation (29).

This establishes a preference of the pattern objective under explicit capacity and similarity conditions. It does not assert that discrete-routing optimization always escapes collapse. For example, take two patches with distinct unit pattern features, $K = 1$, $E \geq 2$, and identical hard assignments. The two rows of S have equal sums, so Equation (28) gives zero routing gradient, even though separate assignments have lower pattern loss. Pattern coherence and optimization dynamics are distinct questions.

C.4 Projection Diversity and the Role of Load Balancing

Learning the metric together with routing makes projection diversity relevant: repeated projection directions reduce the information available for comparing patches. The orthogonality term in Equation (7) penalizes this degeneracy.

Proposition C.4 (rank constraint induced by row orthogonality). Let $m = d_{\text{pat}}$, assume the rows of $\bar{W}_{\text{pat}}$ have unit norm, and let $r = \operatorname{rank}(\bar{W}_{\text{pat}})$. Then

$$\mathcal{R}_{\text{orth}} = \frac{\|\bar{W}_{\text{pat}}\bar{W}_{\text{pat}}^\top - I_m\|_F^2}{m^2} \geq \frac{1}{r} - \frac{1}{m}. \tag{30}$$

In particular, when $m > 1$, a penalty below $1/[m(m-1)]$ implies full row rank, and collinear rows incur a penalty of at least $1 - 1/m$.

Proof. Let $G_W = \bar{W}_{\text{pat}}\bar{W}_{\text{pat}}^\top$. Its trace is m, and it has r positive eigenvalues. Cauchy–Schwarz gives $\|G_W\|_F^2 \geq m^2/r$. Thus,

$$m^2\mathcal{R}_{\text{orth}} = \|G_W\|_F^2 - 2\operatorname{tr}(G_W) + m = \|G_W\|_F^2 - m \geq m^2/r - m.$$

The stated consequences follow by setting $r \leq m - 1$ or $r = 1$.

This result concerns the diversity of projection directions; distinct inputs can still have the same normalized projection. Load balancing addresses a different aspect of routing. The auxiliary loss depends on marginal expert usage and mean routing probabilities, both of which remain unchanged when token-wise assignments and probabilities are jointly permuted. With the patch similarities held fixed, such a permutation can change the pattern loss. Load balancing therefore acts on marginal utilization, while shallow-pattern clustering guides which patches share experts and orthogonality encourages a richer comparison space.