DESA: DISCRETE EVENT-SPARSE ACTIVATION FOR LIGHTWEIGHT MULTIVARIATE TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in multivariate time-series forecasting have primarily focused on improving predictive accuracy. However, these improvements often come at the cost of substantially increased computation, limiting the applicability of such models in resource-constrained industrial environments. In this study, drawing on the event-centric perspective of Discrete Event System Specification (DEVS), we propose DESA (Discrete Event-Sparse Activation), which combines a low-cost base forecast with selective nonlinear correction. The Inertia Block processes the entire input sequence at low computational cost for every forecast, while the Event Block corrects the base forecast using local windows centered at selected time steps. Connecting the two blocks, the Surprise Gate uses reconstruction residuals from the Inertia Block and temporal differences to guide time-step selection and modulate the strength of the Event Block's input. The Event Block gathers the selected time steps for processing and skips its core computations at unselected time steps. To learn this selective processing, we first train a reference model with the same architecture, applying the Event Block at every time step. We then use its predictions and intermediate signals as supervision to train a sparse model that performs selective event processing. Experimental results show that the sparse model achieves an average forecasting error close to that of the reference model while reducing the computational cost of inference. Overall, DESA lies on the empirical forecasting-error–FLOPs Pareto frontier among the compared models, providing a favorable trade-off between forecasting accuracy and computational cost.

1 INTRODUCTION

In long-term multivariate time-series forecasting, accurate predictions are essential in applications such as power grids, transportation networks, weather services, retail, and high-frequency finance. Recent progress from Informer (Zhou et al., 2021) to iTransformer (Liu et al., 2024), and across Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022), Crossformer (Zhang & Yan, 2023), and PatchTST (Nie et al., 2023), has advanced sequence modeling through attention variants and more efficient temporal representations (Wen et al., 2023). Despite these advances, the computational cost of achieving a given forecasting accuracy remains important under limited inference budgets. Figure 1 illustrates this trade-off across the compared models: forward FLOPs differ substantially. Higher computation does not consistently correspond to lower forecasting error in this comparison. This suggests that improving the accuracy–efficiency trade-off requires more than simply increasing computation. We therefore separate low-cost base forecasting from more expensive nonlinear correction. The Inertia Block processes the entire input sequence, while a learned time-step selector determines where the Event Block applies nonlinear correction. Our objective is therefore to reach the empirical MSE–FLOPs Pareto frontier by learning where to allocate additional nonlinear computation, rather than minimizing forecasting error or computation in isolation.

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

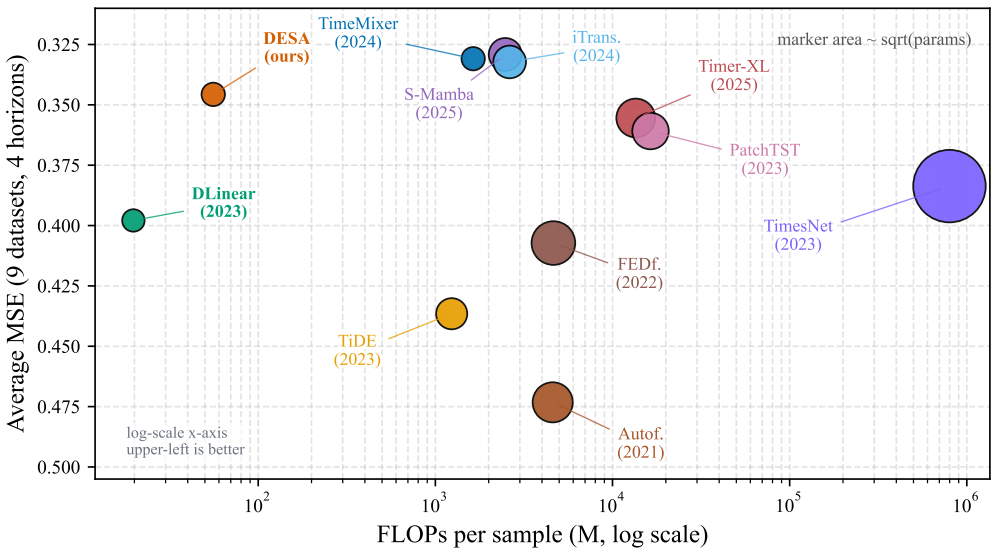


Figure 1: Accuracy–efficiency trade-off. The upper-left region is preferable. Marker area scales with the square root of parameter count.

Low-cost base forecasting and selective correction. Efficient linear forecasters illustrate the value of a low-cost base forecast. DLinear (Zeng et al., 2023) applies linear predictors to decomposed input components and remains competitive on standard long-term forecasting benchmarks at low computational cost. These results support using a low-cost Inertia Block, without restricting it to a linear model. The design does not assume that the input series is stationary.

Time-varying patterns can nevertheless challenge a fixed mapping from historical observations to future values (Liu et al., 2022b). Rather than applying nonlinear correction at every time step, we learn which time steps are most likely to benefit from it. DESA (Discrete Event-Sparse Activation) therefore combines a low-cost base forecast over the entire input sequence with nonlinear correction at selected time steps.

A discrete-event view of forecasting. Discrete Event System Specification (DEVS) (Zeigler, 1976; Zeigler et al., 2000) describes state transitions driven by internal or external events. DESA adopts this event-centric perspective as a design principle for allocating forecasting computation, rather than implementing the DEVS simulation formalism itself. The Inertia Block is always executed, while the time-step selector learns where to apply additional correction. These selections are learned from the forecasting objective, rather than provided as event annotations.

DESA: selective event processing. DESA combines an always-executed Inertia Block with a selectively executed Event Block. Reconstruction residuals and temporal differences form a continuous Surprise Gate (S-Gate) that modulates Event Block inputs and provides evidence to the time-step selector. Its channel-shared binary mask, rather than S-Gate magnitude, determines which time steps serve as centers for Event Block computation. Neighboring observations remain available as local context. A reference model with all centers active guides the sparse model through knowledge distillation of predictions and intermediate signals. Our contributions are:

- We propose DESA, an inertia–correction forecasting architecture that separates low-cost prediction over the entire input sequence from selective nonlinear correction, allowing computation to vary with the input without discarding unselected observations from the base forecast (Section 3).
- We develop a sparse Event Block that combines continuous gate-based input modulation with discrete time-step selection. Packed local-context processing and channel mixing through latent tokens enable cross-channel correction at selected centers while skipping per-center Event computation at unselected centers (Section 3.4).

- We introduce a reference-guided training recipe that transfers predictions and intermediate signals from the reference model to the sparse model with the same architecture. Staged optimization jointly trains forecasting and selection, while inference requires only the sparse model (Section 3.5).

We evaluate these contributions on nine benchmarks across four horizons through accuracy and efficiency comparisons, together with ablations of Event correction, reference-guided training, and time-step selection (Section 4).

2 Related work

Efficient multivariate time-series forecasting. DLinear and NLinear (Zeng et al., 2023) use linear forecasting with input decomposition and last-value normalization, respectively. Their competitive performance at low computational cost supports our motivation for a low-cost base path in the proposed inertia–correction decomposition, implemented by the Inertia Block. TimeMixer (Wang et al., 2024) builds a multi-scale MLP-mixer over downsampled copies of the input. TiDE (Das et al., 2023) is a residual MLP with encoder–decoder blocks. TimesNet (Wu et al., 2023) uses FFT to identify dominant periods, reshapes the sequence into period-aligned 2-D tensors, and models them with 2-D convolutions. FITS (Xu et al., 2024) and FilterNet (Yi et al., 2024) explore frequency-domain modeling for long-term forecasting. FreDF (Wang et al., 2025a) instead introduces a frequency-domain training objective that can be used with different forecasting architectures. SCINet (Liu et al., 2022a) stacks interactive sample-convolution blocks. S-Mamba (Wang et al., 2025b) adapts the selective-scan state-space model into multivariate forecasting. These approaches use diverse mechanisms for temporal and channel modeling. In contrast, DESA focuses on separating base forecasting over the entire input sequence from input-dependent execution of the Event Block.

Transformer forecasters. Informer (Zhou et al., 2021) introduced ProbSparse attention. Autoformer (Wu et al., 2021) and FEDformer (Zhou et al., 2022) use autocorrelation-based and frequency-domain mechanisms, respectively, to model temporal dependencies. PatchTST (Nie et al., 2023) applies a shared Transformer to channel-independent patches, reducing the temporal token count. Crossformer (Zhang & Yan, 2023) models temporal and cross-channel dependencies in two stages, using routers for cross-channel information exchange. iTransformer (Liu et al., 2024) embeds each variate's input history as a token and applies attention across variates. Our Event Block also treats channels as tokens, but only within gathered local contexts. It exchanges information through a bounded set of latent tokens rather than constructing a full channel-attention matrix. DESA uses this channel mixing to form local corrections at selected centers, while an always-executed Inertia Block supplies the base forecast over the entire input sequence.

Discrete-event systems. DEVS formalizes internal and external event-driven state transitions (Zeigler, 1976; Zeigler et al., 2000). Internal transitions occur after a specified time advance, whereas external transitions respond to incoming events. DESA adopts DEVS's event-centric perspective for input-dependent allocation of forecasting computation, rather than implementing the DEVS simulation formalism itself. In DESA, an event is not a predefined simulator event but a time step selected by the model as the center of a learned local correction.

Conditional computation. Early-exit networks (Teerapittayanon et al., 2016) skip the remaining layers once an intermediate classifier produces a sufficiently confident prediction, reducing computation for inputs with high-confidence predictions. Dynamic token pruning in vision Transformers (Rao et al., 2021) drops tokens inside attention based on a learned scorer. DESA instead always executes the Inertia Block and selectively executes the Event Block, using reconstruction residuals and temporal differences as evidence for time-step selection.

3 METHOD

3.1 OVERVIEW

Let $\mathbf{X} \in \mathbb{R}^{B\times L\times C}$ denote the input history and $\mathbf{Y} \in \mathbb{R}^{B\times H\times C}$ the forecast target. We write $\widetilde{\mathbf{X}} = \mathcal{N}_{\mathbf{X}}(\mathbf{X})$ for RevIN normalization (Kim et al., 2022), using statistics from the input history. The Inertia Block produces a reconstruction $\widehat{\mathbf{X}}_{\mathrm{I}}$ and a base forecast $\widehat{\mathbf{Y}}_{\mathrm{I}}$ in this normalized space. The Event Block supplies a selective correction:

$$\widehat{\mathbf{Y}} = \mathcal{N}_{\mathbf{X}}^{-1}\left(\widehat{\mathbf{Y}}_{\mathrm{I}} + |\mathbf{a}| \odot \mathcal{E}(\widetilde{\mathbf{X}}, \mathbf{g}, \mathbf{m})\right), \tag{1}$$

where $\mathbf{a} \in \mathbb{R}^{C}$ is a learned channel-wise output scale, $\mathbf{g} \in (0,1)^{B\times L\times C}$ is the continuous Surprise Gate, and $\mathbf{m} \in \{0,1\}^{B\times L}$ selects time steps that serve as centers of local context windows, referred to as selected centers. Here $\odot$ denotes element-wise multiplication, with channel scales broadcast over the batch and forecast dimensions. The Inertia Block, S-Gate, and time-step selector process the entire input sequence. Per-center Event computation comprises context projection and local channel mixing and runs only at selected centers during inference. An unselected time step can still provide local context for a selected neighbor, but does not undergo its own per-center Event computation. We train a *reference model* with the same architecture and all centers active, then use knowledge distillation to train the *sparse model*: the reference model's predictions and intermediate signals supervise learning of time-step selection and Event correction (Section 3.5). Only the sparse model is used at inference, as illustrated in Figure 2.

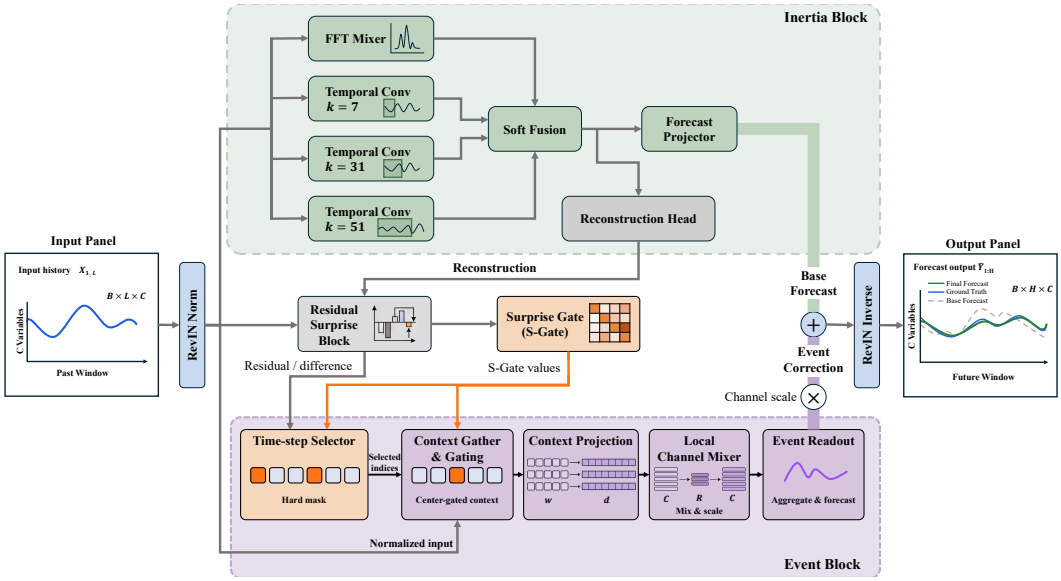


Figure 2: DESA inference architecture.

3.2 INERTIA BLOCK

The channel-wise Inertia Block combines a spectral path (real FFT, learned complex frequency filter, and inverse FFT) with three depth-wise temporal convolutions of kernel sizes 7, 31, and 51. A learned channel-wise softmax mixes the four paths, followed by GELU. Two temporal linear projections of the fused representation produce $\widehat{\mathbf{X}}_{\mathrm{I}}$ and $\widehat{\mathbf{Y}}_{\mathrm{I}}$. The projections are shared across channels. The spectral path costs $\mathcal{O}(CL\log L)$ per sample. The complete block additionally includes convolutions and length-wise projections. It is always executed, including when no centers are selected. The reconstruction head has no separate reconstruction loss. It is optimized through the downstream gate-dependent objectives.

3.3 SURPRISE GATE AND TIME-STEP SELECTION

We omit the batch index when describing a single sample. The two time-channel cues are the reconstruction residual and the absolute first difference:

$$r_{t,c} = \widetilde{x}_{t,c} - \widehat{x}_{\mathrm{I},t,c}, \qquad s_{t,c} = |r_{t,c}|, \tag{2}$$

$$d_{t,c} = |\widetilde{x}_{t,c} - \widetilde{x}_{t-1,c}| \quad (t > 1), \qquad d_{1,c} = 0. \tag{3}$$

The S-Gate combines these cues using a channel-wise two-way softmax:

$$g_{t,c} = \sigma(\kappa\,[\alpha_c s_{t,c} + (1-\alpha_c) d_{t,c} - |\theta_c|])\,, \qquad (\alpha_c, 1-\alpha_c) = \mathrm{softmax}(u_c, v_c), \tag{4}$$

where σ is the sigmoid function and the sharpness is fixed at $\kappa = 10$. Both mixture logits and thresholds are learned. The S-Gate modulates Event Block inputs and provides evidence to the time-step selector. It is not a binary selection mask.

The time-step selector summarizes S-Gate values into a scalar e_t and residual and difference magnitudes into a four-dimensional vector $\mathbf{f}_t$ at each time step. Their channel-wise statistics are defined in Appendix A.2. A learned score gives the relaxed selection probability:

$$\ell_t = b_0 + \mathrm{softplus}(w_g)[e_t - \sigma(c_g)] + f_\psi(\mathbf{f}_t) + b_t, \qquad p_t = \sigma(\ell_t / T), \tag{5}$$

where f_ψ is a small MLP, b_0 and b_t are learned global and position biases, and $T > 0$ is the time-step selector's temperature. The learned parameters w_g and c_g set the positive weight and offset of the S-Gate summary e_t. Selection is determined by a separate threshold test at each center, not by a fixed top-k operation:

$$m_t = \mathbf{1}[p_t \geq \tfrac{1}{2}], \qquad \mathcal{S} = \{t : m_t = 1\}, \qquad K = \sum_{t=1}^{L} m_t. \tag{6}$$

Thus K is input-dependent and can range from zero to L. We define the active-step rate as K/L, the fraction of input time steps selected as centers. The binary selection mask is shared across channels. Here $\mathbf{1}[\cdot]$ denotes the indicator function. Neither neighboring masks nor probabilities are expanded to include the context window.

3.4 EVENT BLOCK

For each selected center, the Event Block gathers a zero-padded local window from the normalized history and multiplies it by that center's channel-wise S-Gate. A center-specific projection maps each channel's window to a d-dimensional token. We use a context radius of two and $d = 64$.

A local channel mixer uses learned queries to pool the C channel tokens into a bounded number of latent tokens that summarize channel information. An MLP transforms these tokens, which are decoded back to the channel tokens through a residual connection. Here, local refers to the temporal context at a center, not a restriction to neighboring channels. This permits cross-channel interaction without constructing a $C \times C$ attention matrix. For fixed latent count and feature width, its cost is linear in C. The selected center tokens are scaled by learned position-dependent weights, summed, and normalized by $\sqrt{K}$. A post-aggregation channel mixer and a shared horizon projection then produce the Event correction.

At inference, contexts are packed only for selected centers, and the post-aggregation channel mixer and horizon projection run only on samples with $K > 0$. When $K = 0$, the correction is exactly zero and the model returns the base forecast. Projection, mixing, and aggregation equations are given in Appendix A.2. The cost decomposition and profiling protocol are given in Appendix B.

3.5 TRAINING RECIPE

Knowledge distillation transfers a trained model's predictive behavior to a model intended for efficient inference (Hinton et al., 2015). Here, the reference model and sparse model share the same architecture. The distinction is how many centers undergo Event computation, rather than their network width or depth.

Train the reference model. We first train the same architecture with all centers active using forecasting MSE. Its S-Gate remains learned: activating all centers bypasses time-step selection, not input modulation. This lets the Event Block learn before introducing the additional task of selecting where to spend computation.

Transfer the correction behavior. We freeze the reference model, copy its parameters other than the time-step selector to the sparse model, and initialize a new time-step selector. In addition to MSE against the observed targets, we minimize MSE between the two models' forecasts. Following the idea of intermediate supervision in FitNets (Romero et al., 2015), we also match magnitude-normalized Event contributions and continuous S-Gate values. These terms guide the Event Block and its input modulation, which matching only the combined forecast would not constrain directly. The reference model supplies regression targets, not class probabilities or a prescribed selection mask.

Introduce selection gradually. We first warm up the Event parameter group with all centers active and the time-step selector frozen, then train both together, and finally fine-tune all model parameters jointly. A straight-through estimator (Bengio et al., 2013) uses binary selection masks in the forward pass and relaxed probabilities for time-step selector gradients. These straight-through batches evaluate Event features at all centers. The packed inference path evaluates them only at selected centers. We gradually introduce a computation penalty alongside the forecasting and distillation losses, allowing the time-step selector to trade computational cost against the corrections needed for prediction. Validation error and computation determine the selected checkpoint. Only the sparse model is used at inference. Loss definitions, stage settings, and the checkpoint rule are provided in Appendix A.3.

4 Experiments

4.1 Experimental setup

We evaluate on nine standard long-term benchmarks—Weather, Solar-Energy, Electricity, Traffic, Exchange, and ETTh1/ETTh2/ETTm1/ETTm2 (Zhou et al., 2021; Wu et al., 2021)—with input length L=96 and horizons $H \in \{96, 192, 336, 720\}$. We compare against the nine baselines listed in Table 2. We use published baseline results and supplement them with evaluations using official implementations. DESA reports mean accuracy and sample standard deviation using the aggregation procedure in Appendix A.5. Forward operation counts in Table 1 and Figure 1 are counted with a PyTorch FLOPs counter (Paszke et al., 2019) at batch size 1. Runtime measurements follow the protocol in Appendix B. For DESA, we use the same training recipe across datasets and horizons. The settings are listed in Appendix A.

4.2 Efficiency

DESA uses an average of 55.74 M operator-counted FLOPs and $146{,}018$ parameters (Table 1). The FLOP counts reflect actual sparse execution. Figure 3 relates forecasting error to operation counts and measured inference time, using logarithmic cost axes as in Figure 1. FLOPs and parameters average nine datasets and four horizons. Runtime and memory average nine datasets at $L = H = 96$, batch size 16. The MSE in each panel uses the same forecast horizons as its cost metric. Profiling differences are detailed in Appendix B.

4.3 Forecasting accuracy

Forecasting accuracy provides the context for interpreting the computational savings. Table 2 compares DESA with linear, convolutional, mixing, attention, and state-space baselines, including Timer-XL (Liu et al., 2025). DESA obtains overall MSE 0.346 and MAE 0.355, with performance varying across datasets rather than uniformly leading the comparison. Together with the efficiency results, these values characterize the accuracy–cost trade-off of selective correction. The full 36-condition results are in Appendix F.

Table 1: Computational cost and inference efficiency. Parenthesized ratios are relative to DESA.

Model	FLOPs (M)	Params	CPU time (ms/iter)	GPU time (ms/iter)	GPU memory (MiB)
DESA (ours)	55.74 (1.00×)	146,018 (1.00×)	20.40	3.52	107.40
Timer-XL	13,529.70 (242.72×)	9,406,583 (64.42×)	146.7	11.85	335.4
S-Mamba	2,479.69 (44.48×)	3,576,850 (24.50×)	867.66	8.51	75.6
TimeMixer	1,635.31 (29.34×)	137,903 (0.94×)	123.3	11.90	215.0
iTransformer	2,625.75 (47.11×)	3,391,710 (23.23×)	91.7	7.60	200.1
PatchTST	16,387.47 (293.99×)	6,627,778 (45.39×)	132.4	19.88	161.8
Crossformer	171,664.17 (3079.60×)	31,527,584 (215.92×)	890.8	60.49	312.3
TiDE	1,236.50 (22.18×)	2,444,398 (16.74×)	2219.4	163.70	52.5
TimesNet	798,305.63 (14321.34×)	184,492,823 (1263.49×)	239.7	16.07	57.9
DLinear	19.74 (0.35×)	65,184 (0.45×)	0.7	0.27	15.9

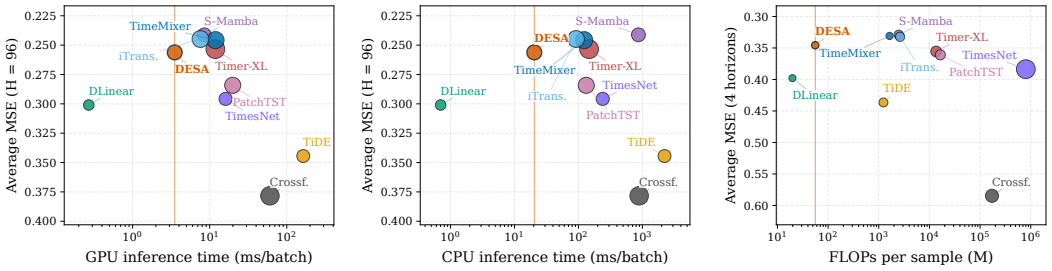


Figure 3: Accuracy–efficiency trade-offs. MSE averages nine datasets at $H = 96$ in the time panels and over four horizons in the FLOPs panel. The upper-left region is preferable. Bubble sizes encode GPU memory in both time panels and parameter count in the FLOPs panel.

Table 2: Forecasting results averaged over $H \in \{96, 192, 336, 720\}$ ($L = 96$). DESA: mean (± SD). Red/blue bold indicate first/second place, including ties.

	DESA (ours)		Timer-XL (2025)		S-Mamba (2025)		TimeMixer (2024)		iTransformer (2024)		PatchTST (2023)		Crossformer (2023)		TiDE (2023)		TimesNet (2023)		DLinear (2023)	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	**0.249 (±0.0002)**	0.278 (±0.0001)	0.260	0.281	0.251	**0.276**	**0.240**	**0.272**	0.258	0.278	0.265	0.286	0.264	0.320	0.271	0.320	0.259	0.287	0.265	0.315
Solar-Energy	0.243 (±0.0037)	0.280 (±0.0020)	0.300	0.303	0.240	**0.273**	**0.216**	0.280	**0.233**	**0.262**	0.287	0.333	0.406	0.442	0.347	0.418	0.403	0.374	0.330	0.401
Electricity	0.188 (±0.0015)	0.281 (±0.0019)	**0.174**	**0.267**	**0.170**	**0.265**	0.182	0.273	0.178	0.270	0.216	0.318	0.244	0.334	0.252	0.344	0.193	0.295	0.225	0.319
Traffic	0.504 (±0.0019)	0.346 (±0.0023)	0.541	0.346	**0.414**	**0.276**	0.485	0.298	**0.428**	**0.282**	0.529	0.341	0.667	0.426	0.761	0.473	0.620	0.336	0.625	0.383
Exchange	0.387 (±0.0016)	0.417 (±0.0007)	**0.354**	**0.402**	0.367	0.408	0.387	0.416	**0.360**	**0.403**	0.367	0.404	0.940	0.707	0.370	0.414	0.416	0.443	**0.354**	0.414
ETTh1	0.466 (±0.0048)	0.452 (±0.0042)	**0.442**	**0.438**	0.455	0.450	**0.447**	**0.440**	0.454	0.448	0.503	0.476	0.529	0.522	0.541	0.507	0.458	0.450	0.461	0.458
ETTh2	**0.381 (±0.0011)**	**0.404 (±0.0007)**	0.397	0.412	**0.381**	0.405	**0.365**	**0.395**	0.383	0.407	0.391	0.412	0.942	0.684	0.611	0.550	0.414	0.427	0.563	0.519
ETTm1	0.409 (±0.0019)	0.411 (±0.0013)	0.440	0.428	**0.398**	**0.405**	**0.381**	**0.396**	0.407	0.410	0.401	0.406	0.513	0.495	0.419	0.419	0.400	0.406	0.404	0.408
ETTm2	**0.286 (±0.0011)**	**0.329 (±0.0007)**	0.291	0.333	0.288	0.332	**0.275**	**0.323**	0.288	0.332	0.290	0.334	0.757	0.611	0.358	0.404	0.291	0.333	0.354	0.402
Overall	0.346 (±0.0004)	0.355 (±0.0002)	0.355	0.357	**0.329**	**0.343**	**0.331**	**0.344**	0.332	**0.343**	0.361	0.368	0.585	0.505	0.437	0.428	0.384	0.372	0.398	0.402

4.4 TIME-STEP SELECTION AND FORECAST CORRECTION

Figure 4 illustrates time-step selection and forecast correction on ETTh1 with $L = H = 96$. Orange marks indicate selected centers, and the shaded gap shows the Event correction to the base forecast from the same trained model. In this example, the correction adjusts the forecast amplitude, illustrating how selective Event processing supplements the base forecast computed from the entire input history. Appendix E provides additional examples for all nine datasets and four horizons.

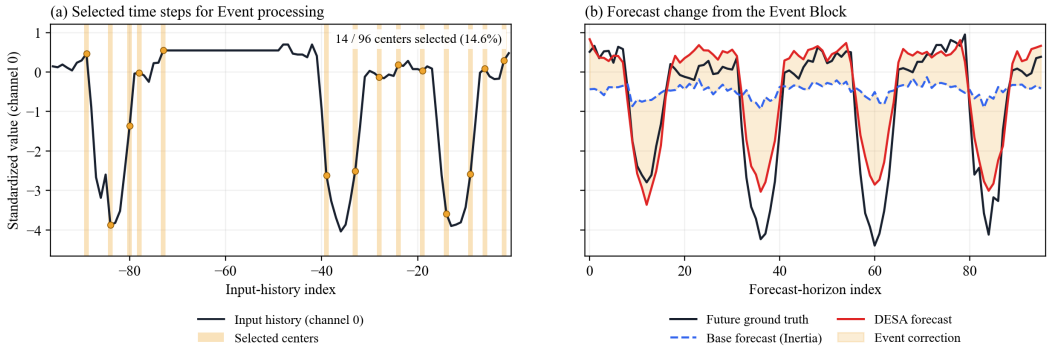


Figure 4: Time-step selection (left) and forecast correction (right) on ETTh1.

4.5 ABLATIONS

The ablations examine whether correction is useful, where it should be applied, and how reference-guided training supports sparse execution. All results in Table 3 average the 36 benchmark conditions. Inertia-only has higher MSE than DESA (0.3647 vs. 0.3457), supporting the contribution of Event correction. The reference model provides a comparison for the accuracy–cost trade-off: the sparse model retains similar mean MSE while reducing FLOPs from 150.90 M to 55.74 M.

Table 3: Training and time-step selection ablations.

Variant	Active steps (%)	MSE	MAE	FLOPs (M)
Reference model	100.00	0.3440	0.3516	150.90
DESA	25.17	0.3457	0.3553	55.74
S-Gate time shuffle	29.58	0.3471	0.3563	75.55
Random matched-K	25.17	0.3631	0.3683	71.05
Teacher-free	12.08	0.3762	0.3719	30.90
Inertia-only	0.00	0.3647	0.3628	12.48

S-Gate time shuffle tests the importance of temporally aligned S-Gate values. It increases both MSE and the active-step rate. Random matched-K tests selection locations while preserving each sample's selected count. Its higher MSE (0.3631) shows that the count alone does not explain DESA's accuracy. Its FLOPs include the operations used to obtain the matched count. Training the sparse model without a reference model also increases MSE (0.3762), suggesting that reference-guided training helps preserve accuracy in the sparse model. The teacher-free model trains without a reference model using its own recipe, rather than removing only distillation losses (Appendix A.4). Full per-condition results are in Appendix C. Sensitivity analyses of input length, Event width, and the computation penalty weight are reported in Appendix D.

5 DISCUSSION

The results support DESA's separation of low-cost base forecasting in the Inertia Block from selective nonlinear correction in the Event Block. Because the Inertia Block already supplies a base forecast, skipping an Event computation does not remove that time step from the entire predictor. Instead,

selection determines where to spend additional computation. This distinction allows DESA to use local context around selected centers without executing per-center Event computation at every context position.

The time-step selection controls further suggest that useful sparsity depends on the alignment and location of correction, not only on the active-step rate. The S-Gate modulates Event Block inputs continuously, whereas the time-step selector makes the discrete execution decision. The reported 25.17% rate measures selected centers, not gate intensity. Reference-guided training uses the reference model's predictions and intermediate signals as targets while the sparse model learns this selective behavior. Only the sparse model is used at inference, so the additional reference training does not add to its inference cost.

Limitations. DESA's selection patterns depend on the input length and computation penalty weight, as examined in Appendix D, and may also vary with the normalization method. The Teacher-free comparison uses a different optimization recipe. FLOPs and parameter counts do not capture dispatch overhead or device utilization. The measured runtime is hardware- and input-dependent. Streaming evaluation and distribution-shift tests remain necessary for assessing deployment.

6 Conclusion

We present DESA, an inertia–correction architecture that integrates input-dependent computation allocation into multivariate time-series forecasting. Its central contribution is to separate using the full input history from applying nonlinear correction at every time step: the Inertia Block uses the entire input sequence to produce a low-cost base forecast, while input-dependent selection determines which Event computations are actually executed. Reference-guided training transfers predictions and intermediate signals from the reference model to the sparse model with the same architecture, with only the sparse model needed at inference. When averaged over nine datasets and four forecasting horizons, DESA lies on the empirical MSE–FLOPs Pareto frontier among the compared models. Within the proposed architecture, the sparse model achieves mean MSE close to the reference model while reducing operator-counted FLOPs from $150.90\,\mathrm{M}$ to $55.74\,\mathrm{M}$. This demonstrates the value of allocating computation according to its role in prediction, rather than reducing cost solely by shrinking a uniformly executed model. The resulting architecture and training recipe offer a foundation for budget-aware forecasting systems that retain historical context while adapting the correction workload to the input. Future work will investigate the use of DESA for real-time and streaming forecasting under resource constraints, explore time-series foundation models based on its inertia–correction architecture, and develop modular approaches for integrating selective event processing into other forecasting architectures.

AI use statement

Generative AI tools assisted with manuscript preparation, including translation. The authors are responsible for the final content and its verification.

References

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with TiDE: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023.

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

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

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

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022b.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.06625.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer-XL: Long-context transformers for unified time series forecasting. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. arXiv:2211.14730.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2106.02034.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In *23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469, 2016.

Hao Wang, Lichen Pan, Yuan Shen, Zhichao Chen, Degui Yang, Yifei Yang, Sen Zhang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. FreDF: Learning to forecast in the frequency domain. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025a.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. arXiv:2405.14616.

Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is Mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025b. doi: 10.1016/j.neucom.2024.129178.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2106.13008.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

Zhijian Xu, Ailing Zeng, and Qiang Xu. FITS: Modeling time series with 10k parameters. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. FilterNet: Harnessing frequency filters for time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Bernard P. Zeigler. *Theory of Modelling and Simulation*. Wiley–Interscience, New York, 1976.

Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, San Diego, CA, 2nd edition, 2000.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023. arXiv:2205.13504.

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

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11106–11115, 2021. arXiv:2012.07436.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 27268–27286, 2022. arXiv:2201.12740.

A REPRODUCIBILITY DETAILS

A.1 DATA AND EVALUATION

Datasets and splits. We use the nine public benchmarks in Table 4, with input length $L = 96$ and horizons $H \in \{96, 192, 336, 720\}$. Data are split chronologically. Dataset standardization uses training-set statistics. ETT uses the first 12 months for training, the next four for validation, and the following four for testing, with each month implemented as 30 days. Other datasets use chronological 70/10/20% splits with integer rounding. Validation and test histories can extend into the preceding partition for context, but targets remain within their respective partitions.

Table 4: Dataset summary. Dim: number of variables. Split: chronological train/validation/test partition.

Dataset	Dim	Frequency	Total length	Split	Domain
ETTh1 / ETTh2	7	1 hour	17,420	12/4/4 months	Electricity transformer
ETTm1 / ETTm2	7	15 min	69,680	12/4/4 months	Electricity transformer
Weather	21	10 min	52,696	7:1:2	Meteorology
Solar-Energy	137	10 min	52,560	7:1:2	Solar power
Electricity	321	1 hour	26,304	7:1:2	Electricity consumption
Traffic	862	1 hour	17,544	7:1:2	Road occupancy
Exchange	8	1 day	7,588	7:1:2	Exchange rates

Evaluation metrics. MSE and MAE are computed in the dataset-standardized space. For DESA, inverse RevIN is applied before evaluating these metrics.

Baseline evaluation. Timer-XL is trained locally with $L = 96$ and uses rolling prediction for longer horizons. For each baseline, dataset-level scores average the four horizon-specific test results.

A.2 ARCHITECTURE DETAILS

Time-step selector statistics. For a vector across channels, let mean_c denote its mean and define the stabilized root mean square (RMS) as $\mathrm{rms}_c(z) = \sqrt{\mathrm{mean}_c(z^2) + 10^{-8}}$. The time-step selector in Eq. (5) uses

$$\begin{aligned} e_t &= \tfrac{1}{2}(\mathrm{mean}_c\, g_{t,c} + \mathrm{rms}_c\, g_{t,c})\,, \\ \mathbf{f}_t &= [\mathrm{mean}_c\, s_{t,c}, \mathrm{rms}_c\, s_{t,c}, \mathrm{mean}_c\, d_{t,c}, \mathrm{rms}_c\, d_{t,c}]\,. \end{aligned} \tag{7}$$

The function f_ψ applies LayerNorm followed by a $4 \to 32 \to 1$ MLP with GELU to $\mathbf{f}_t$. Its output is combined with the positively weighted gate summary e_t in Eq. (5) to obtain a selection probability for each time step. Thresholding these probabilities yields the binary mask and the selected centers used below.

Event implementation. Let $\mathcal{S} = \{t : m_t = 1\}$ be the selected centers, with $K = |\mathcal{S}|$. The gate and mask are defined in Section 3.3. For each selected center $t \in \mathcal{S}$, we gather a window of radius ρ and width $w = 2\rho + 1$ from the normalized history. We define $\widetilde{x}_{k,c} = 0$ for indices k outside $1, \ldots, L$. The center's channel gate is broadcast across this window:

$$\begin{aligned} U_{t,j,c} &= g_{t,c}\, \widetilde{x}_{t+j,c}, \qquad -\rho \le j \le \rho, \\ Z_{t,c,:} &= \sum_{j=-\rho}^{\rho} U_{t,j,c} W_{t,:,j+\rho+1} + \mathbf{q}_t, \end{aligned} \tag{8}$$

where $W_t \in \mathbb{R}^{d \times (2\rho+1)}$ and $\mathbf{q}_t \in \mathbb{R}^d$ are center-specific projection weights and position embeddings shared across channels. The offset j indexes relative time, and $j + \rho + 1$ indexes the corresponding column of W_t. A colon denotes all entries along that dimension, and $Z_t \in \mathbb{R}^{C \times d}$. We use radius

$\rho = 2$ and Event width $d = 64$. The gate $g_{t,c}$ is used throughout the window, not the neighbor's gate $g_{t+j,c}$. Only K such contexts are gathered and processed during inference.

The projected features next undergo channel mixing through a bounded set of latent tokens. The local channel mixer $\mathcal{M}_{\text{loc}}$ and post-aggregation channel mixer $\mathcal{M}_{\text{post}}$ share this form but have separate parameters. For tokens $Z \in \mathbb{R}^{C\times d}$, each mixer uses R learned pooling and decoding queries $Q_{\text{p}}, Q_{\text{d}} \in \mathbb{R}^{R\times d}$:

$$\begin{aligned} N &= \text{LN}_{\text{in}}(Z), & P &= \text{softmax}_C(NQ_{\text{p}}^\top/\sqrt{d}), \\ Z_{\text{lat}} &= P^\top Z, & \widetilde{Z}_{\text{lat}} &= Z_{\text{lat}} + \text{MLP}(\text{LN}_{\text{lat}}(Z_{\text{lat}})), \\ D &= \text{softmax}_R(NQ_{\text{d}}^\top/\sqrt{d}), & \mathcal{M}(Z) &= Z + \tanh(\eta)D\widetilde{Z}_{\text{lat}}. \end{aligned} \tag{9}$$

Here $Z_{\text{lat}} \in \mathbb{R}^{R\times d}$ contains the pooled latent tokens and η is a learned scalar. Each mixer uses separate input and latent LayerNorms (LN_{in} and LN_{lat}) and a Linear–GELU–Linear MLP. Pooling normalizes over channels, while decoding normalizes over latents. The pooling values are the original tokens Z, not N. No $C \times C$ attention matrix is formed. With MLP hidden width d_{ff}, each mixer costs $\mathcal{O}(CRd + Rdd_{\text{ff}})$. This is linear in the channel count C when widths and the number of latent tokens R are fixed. We use the bounded counts $R_{\text{loc}} = \min(16, \max(4, \lceil C/16 \rceil))$ and $R_{\text{post}} = \min(32, \max(8, 2R_{\text{loc}}))$. The local and post MLP widths are 32 and 64, respectively.

Finally, local features are scaled and aggregated across selected centers, then passed through the post-aggregation channel mixer and horizon projection. Let $\mathbf{v}_t \in \mathbb{R}^d$ be a learned center-specific token scale. For $K > 0$, the Event correction is

$$\begin{aligned} V_t &= \mathcal{M}_{\text{loc}}(Z_t) \odot \mathbf{v}_t, \qquad A = \frac{1}{\sqrt{K}} \sum_{t\in\mathcal{S}} V_t + \mathbf{b}_{\text{s}}, \\ \mathcal{E}(\widetilde{\mathbf{X}}, \mathbf{g}, \mathbf{m}) &= \left(\mathcal{M}_{\text{post}}(A)W_{\text{H}}^\top + \mathbf{b}_{\text{H}}\right)^\top, \end{aligned} \tag{10}$$

where $A \in \mathbb{R}^{C\times d}$, $\mathbf{b}_{\text{s}} \in \mathbb{R}^d$, $W_{\text{H}} \in \mathbb{R}^{H\times d}$, and $\mathbf{b}_{\text{H}} \in \mathbb{R}^H$. Biases and token scales broadcast over channels. If $K = 0$, $\mathcal{E} = 0$ exactly and the complete Event Block, including its post-aggregation channel mixer and horizon projection, is bypassed during sparse inference. For a batch, selected centers are packed together. The post-aggregation channel mixer and horizon projection run only on samples with at least one selected center.

A.3 Training details

Table 5: Training hyperparameters.

Hyperparameter	Reference model	Sparse model (DESA)
Input length (L)	96	96
Event width (d)	64	64
Training epochs (maximum)	113	24
Forecast horizon (H)	96, 192, 336, 720	96, 192, 336, 720
Effective batch size	32	32
Context radius (ρ)	2	2
Computation penalty (γ)	–	0.1
Distillation weights ($\lambda_{\text{p}}, \lambda_{\text{e}}, \lambda_{\text{g}}$)	–	1, 1, 0.1
Optimizer	AdamW	AdamW
Backbone learning rate	10^{-4}	10^{-5}
Event parameter group learning rate	3×10^{-4}	10^{-3}
Time-step selector learning rate	–	3×10^{-3}
Weight decay	10^{-4}	10^{-5}
Gradient norm limit	1.0	1.0
Learning-rate schedule	$\times 0.25$ after epoch 83	Fixed

Losses and optimization. Let superscripts S and R denote outputs of the sparse model and frozen reference model. We use element-averaged MSE, denoted by MSE, and train with

$$\begin{aligned}\mathcal{L} = \ & \mathrm{MSE}(\widehat{\mathbf{Y}}^{\mathrm{S}}, \mathbf{Y}) + \lambda_{\mathrm{p}} \mathrm{MSE}(\widehat{\mathbf{Y}}^{\mathrm{S}}, \widehat{\mathbf{Y}}^{\mathrm{R}}) \\ & + \lambda_{\mathrm{e}} \mathcal{L}_{\mathrm{event}} + \lambda_{\mathrm{g}} \mathrm{MSE}(\mathbf{g}^{\mathrm{S}}, \mathbf{g}^{\mathrm{R}}) + \gamma \, s_{\mathrm{stage}} \mathcal{R}_{\mathrm{compute}}.\end{aligned} \tag{11}$$

Here $\mathcal{L}_{\mathrm{event}}$ aligns magnitude-normalized Event contributions, and $\mathcal{R}_{\mathrm{compute}}$ penalizes expected Event computation. We use $\lambda_{\mathrm{p}} = \lambda_{\mathrm{e}} = 1$, $\lambda_{\mathrm{g}} = 0.1$, and $\gamma = 0.1$. The multiplier s_{stage} introduces the computation penalty during training.

Forecast losses are evaluated after inverse RevIN, in the same dataset-standardized space as the target. For Event supervision, let $E_b^{\mathrm{S}}, E_b^{\mathrm{R}} \in \mathbb{R}^{H \times C}$ be the additive Event contributions in that same space, including the learned channel scale and inverse RevIN scaling, but no additive normalization offset. Define $\mathrm{rms}(E) = \sqrt{\mathrm{mean}(E^2)}$. The normalized Event loss is

$$\begin{aligned} a_b^{\mathrm{KD}} &= \max\left\{\mathrm{rms}(E_b^{\mathrm{R}}),\ 0.1\,\mathrm{rms}(E^{\mathrm{R}}),\ 10^{-4}\right\}, \\ \mathcal{L}_{\mathrm{event}} &= \frac{1}{BHC} \sum_{b,h,c} \left(\frac{E_{b,h,c}^{\mathrm{S}} - E_{b,h,c}^{\mathrm{R}}}{a_b^{\mathrm{KD}}} \right)^2 . \end{aligned} \tag{12}$$

The RMS in the floor term is computed over the current minibatch. All reference quantities are detached. Gate supervision matches continuous S-Gate values, not binary selection masks or time-step selection probabilities.

To penalize computation during training, we use a differentiable approximation to the Event Block's computational cost, measured in multiply-accumulate operations (MACs). We normalize this estimate by the cost of processing all L centers:

$$\mathcal{R}_{\mathrm{compute}} = \frac{1}{B} \sum_{b=1}^{B} \frac{c_{\mathrm{ctr}} \sum_{t=1}^{L} p_{b,t} + c_{\mathrm{samp}} \left[1 - \prod_{t=1}^{L} (1 - p_{b,t}) \right]}{L c_{\mathrm{ctr}} + c_{\mathrm{samp}}}, \tag{13}$$

where c_{ctr} counts context projection and local mixing per center, and c_{samp} counts the post-aggregation channel mixer and horizon projection per active sample. With local/post MLP widths d_ℓ, d_{p}, these costs are approximated by

$$\begin{aligned} c_{\mathrm{ctr}} &= C(2\rho + 1)d + 4CR_{\mathrm{loc}}d + 2R_{\mathrm{loc}}dd_\ell, \\ c_{\mathrm{samp}} &= 4CR_{\mathrm{post}}d + 2R_{\mathrm{post}}dd_{\mathrm{p}} + CdH. \end{aligned} \tag{14}$$

The sum of selection probabilities estimates the number of active centers. The term $1 - \prod_t (1 - p_{b,t})$ is the probability of at least one active center under independent Bernoulli selection. We use it only as a differentiable approximation: inference thresholds probabilities deterministically and does not sample masks. In implementation, probabilities in that product are capped at $1 - 10^{-7}$ for stability. The cost estimate in Eqs. (13)–(14) includes the main matrix operations but excludes the always-executed branches, element-wise operations, and data movement. It is distinct from the operator-counted FLOPs reported in Table 1.

Straight-through training. For training with the straight-through estimator (Bengio et al., 2013), we use $\widetilde{m}_t = p_t + \mathrm{sg}(m_t - p_t)$, where sg is the identity in the forward pass and has zero gradient in the backward pass. The forward selection is binary, while backward gradients use the relaxed probability. These training batches compute Event features at all centers, whereas sparse inference gathers only selected centers. The sample-active indicator is treated analogously: with $q_{\mathrm{any}} = 1 - \prod_{t=1}^{L} (1 - p_t)$, define $\widetilde{u} = q_{\mathrm{any}} + \mathrm{sg}(\mathbf{1}[K > 0] - q_{\mathrm{any}})$. The vectorized training aggregation is

$$\begin{aligned} \widetilde{A} &= \frac{\sum_{t=1}^{L} \widetilde{m}_t V_t}{\sqrt{\max(1, \sum_{t=1}^{L} \widetilde{m}_t)}} + \widetilde{u}\, \mathbf{b}_{\mathrm{s}}, \\ \widetilde{\mathcal{E}} &= \widetilde{u} \left(\mathcal{M}_{\mathrm{post}}(\widetilde{A}) W_{\mathrm{H}}^{\top} + \mathbf{b}_{\mathrm{H}} \right)^{\top}. \end{aligned} \tag{15}$$

In Eq. (15), gradients pass through the relaxed count in the denominator. The final indicator multiplication preserves a zero Event output for unselected samples, including after the mixer and output biases have been learned.

Training stages and checkpoint selection. The reference model minimizes forecasting MSE for up to 113 epochs and uses the checkpoint with minimum validation MSE with all centers active. At epoch 84, AdamW is reinitialized with learning rates reduced to one quarter. All parameters of the reference model except the unused time-step selector are trainable. The sparse model uses three stages: 8 epochs of Event parameter group warm-up (stage 1), 10 epochs of joint Event parameter group and time-step selector training (stage 2), and 6 epochs of fine-tuning all parameters (stage 3). Optimizer settings are summarized in Table 5. Warm-up executes all centers with the time-step selector frozen. During joint fine-tuning, every fourth minibatch uses sparse execution. The others use the straight-through estimator. The computation penalty multiplier is zero in stage 1, $(j-1)/9$ at epoch $j = 1, \ldots, 10$ of stage 2, and one in stage 3. The time-step selector's temperature is geometrically annealed from 1.5 to 0.35 over stages 2 and 3. Inference retains the selected checkpoint's temperature. Binary and contiguity penalties are disabled. The Event parameter group contains the Event Block parameters except the channel output scale, and is updated during warm-up and time-step selector training. The backbone comprises the remaining non-selector parameters. The channel output scale belongs to the backbone group when training the sparse model (updated in stage 3) and to the Event parameter group when training the reference model.

For each training run, we select a checkpoint of the sparse model using the validation criteria below. The checkpoint rule jointly considers validation error and computation. Candidates within 5% of the reference in mean MSE and reference-defined worst-10% tail MSE are ranked by analytic MACs, then MSE. The tail comprises the validation samples with the largest MSE of the reference model, evaluated over the same sample indices for both models. If no candidate is feasible, the sum of positive relative constraint violations is minimized, followed by MSE and then analytic MACs.

A.4 Ablation controls

Inertia-only control. The Inertia Block and RevIN are trained from scratch without the Event Block, S-Gate, time-step selector, or reference supervision. The unused reconstruction head is omitted. Training uses forecast MSE, AdamW with learning rate 10^{-4} and weight decay 10^{-4}, and the reference model's 113-epoch budget (83 epochs followed by 30 at one quarter of the learning rate). Effective batch size is 32. Minimum validation MSE selects the checkpoint.

Teacher-free control. The Teacher-free comparison initializes the sparse model from scratch without reference supervision. It uses forecasting MSE and a computation penalty with $\gamma = 0.2$ over up to 113 epochs: 3 epochs of warm-up with all centers active, 80 of joint straight-through training, and 30 with the time-step selector frozen. Backbone and Event/time-step selector learning rates are 10^{-4} and 3×10^{-4}, respectively. The final stage uses one quarter of these rates. Checkpoints minimize validation MSE plus 0.2 times the executed-to-full Event MAC ratio. This comparison evaluates training without a reference model under its own recipe, rather than isolating reference supervision with all other settings held fixed.

Time-step selection controls. S-Gate time shuffle and Random matched-K each retrain 36 sparse models using the fixed reference models and the three-stage recipe, including all-center warm-up. Both controls are retrained, rather than applied only at inference.

S-Gate time shuffle applies an input-history-keyed pseudorandom time permutation to S-Gate values used by the time-step selector and Event Block input modulation, including warm-up. Shared across channels and reused for the same input, it preserves each channel's gate-value distribution. Residual and difference features remain aligned. Gate supervision uses the unshuffled S-Gate.

Random matched-K obtains each input's selected-center count from the frozen original sparse model, rather than the reference model, then selects that many positions without replacement using an input-history-keyed permutation and no target values. Its S-Gate remains aligned, while its time-step selector is frozen and unused. The matched mask is applied after warm-up and at every hard evaluation. The retained computation penalty therefore supplies no trainable time-step selection gradient. Count-acquisition overhead is included in the FLOPs accounting in Appendix B.

A.5 SEED VARIATION AND EXPERIMENTAL ENVIRONMENT

Reporting scope. Reference models are trained with seed 2021. DESA accuracy in Tables 2 and 13 uses seeds 2021, 2022, and 2023 for the sparse model, sharing the same 36 fixed reference checkpoints, one per condition. Ablations, sensitivity studies, and efficiency comparisons use seed 2021. Efficiency plots pair accuracy and computation from that same seed. Qualitative time-step selection figures use seed 2022.

Randomness and data loading. The designated seed is applied to Python, NumPy, PyTorch, and CUDA. Training data are shuffled with incomplete final batches dropped. Validation and test data are not shuffled. Data loaders use zero worker processes. For seeds 2022 and 2023, cuDNN benchmarking and deterministic mode are both disabled.

Training and profiling environment. Training uses single-GPU CUDA runs. The separate profiling environment is described in Appendix B.

Seed variation. Tables 6 and 7 report results for the sparse model across these three seeds. Per-condition results report the mean and sample standard deviation across seeds. Each dataset is averaged over four horizons within a seed, and each overall value over 36 conditions. Means and sample standard deviations ($n = 3$, denominator $n - 1$) are then computed across these seed-level averages. The reported variation concerns training of the sparse model with fixed reference models, not the full training pipeline. Ablations use a single seed.

Table 6: Seed variation of the sparse model with fixed reference models, averaged over 36 conditions.

Seed	MSE	MAE
2021	0.345668	0.355270
2022	0.345766	0.355232
2023	0.346391	0.355682
Mean $\pm$ SD	0.345942 ± 0.000392	0.355395 ± 0.000249

Table 7: Dataset-level seed variation of the sparse model with fixed reference models.

	2021		2022		2023		Mean $\pm$ SD	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	0.2491	0.2780	0.2488	0.2778	0.2491	0.2780	0.2490 ± 0.0002	0.2779 ± 0.0001
Solar-Energy	0.2390	0.2778	0.2430	0.2798	0.2464	0.2818	0.2428 ± 0.0037	0.2798 ± 0.0020
Electricity	0.1892	0.2826	0.1863	0.2789	0.1880	0.2813	0.1878 ± 0.0015	0.2809 ± 0.0019
Traffic	0.5051	0.3476	0.5045	0.3474	0.5016	0.3435	0.5038 ± 0.0019	0.3461 ± 0.0023
Exchange	0.3885	0.4181	0.3873	0.4176	0.3854	0.4167	0.3871 ± 0.0016	0.4175 ± 0.0007
ETTh1	0.4622	0.4486	0.4648	0.4507	0.4716	0.4566	0.4662 ± 0.0048	0.4520 ± 0.0042
ETTh2	0.3803	0.4034	0.3803	0.4035	0.3822	0.4047	0.3810 ± 0.0011	0.4039 ± 0.0007
ETTm1	0.4098	0.4112	0.4111	0.4123	0.4074	0.4097	0.4094 ± 0.0019	0.4111 ± 0.0013
ETTm2	0.2878	0.3301	0.2858	0.3291	0.2858	0.3289	0.2865 ± 0.0011	0.3294 ± 0.0007

B EFFICIENCY ACCOUNTING DETAILS

Execution cost. The binary selection mask, rather than the mean S-Gate value, determines where per-center Event computation runs. Let $N_{\text{sel}} = \sum_{b=1}^{B} K_b$ count selected centers and $N_{\text{act}} = \sum_{b=1}^{B} \mathbf{1}[K_b > 0]$ count samples with at least one selected center. An arithmetic-cost decomposition is

$$F_{\text{batch}} = BF_{\text{front}} + N_{\text{sel}}F_{\text{center}} + N_{\text{act}}F_{\text{head}} + F_{\text{other}}. \tag{16}$$

The front-end term includes always-executed normalization, Inertia, S-Gate, and time-step selection. The center term includes context projection and the local channel mixer. The head term includes the post-aggregation channel mixer and horizon projection. F_{other} covers additional arithmetic in time-step selection, aggregation, and fusion. Data movement also incurs runtime overhead. This decomposition describes computation, not wall-clock latency.

FLOPs accounting. FLOPs are operator counts, not latency estimates. Sparse counts are weighted by the observed test-time selection cardinalities, and parameters of the reference model are excluded from inference with the sparse model. Unsupported operations such as FFT and element-wise operations may be omitted. Baseline costs retain their recorded profiling settings. Table 3 uses the same operator counter at batch size 1. Random matched-K includes the frozen baseline's encoding and time-step selection operations needed to obtain each sample's count. Input hashing and random permutation are not counted as floating-point operations.

Runtime and memory. Runtime uses $H = 96$, batch size 16, eight CPU threads, and an RTX 5060 Ti, with FP32 and gradients disabled. For each dataset, 16 evenly spaced complete test batches are materialized before timing. Each batch has five warm-up and five timed forward passes. CUDA is synchronized at timing boundaries. We take the median of three replicate batch means, then average equally across nine datasets. Peak CUDA allocated memory includes the model and one resident input batch, but not the CUDA context. Loading and device transfer are excluded. Unlike the retained random-input baseline profiles, DESA uses real test inputs because its execution depends on the selected time steps. S-Mamba's CPU latency is measured using a reference implementation because its native kernel requires CUDA.

Table 8: FLOPs counting protocol.

Item	DESA
Counter	PyTorch FlopCounterMode
Mode	Evaluation without gradients
Profiling batch size	1
Input length	96
Aggregation	36 conditions with observed test selection counts
Reference cost	Excluded from sparse inference
Unsupported operations	Not guaranteed to be counted

C FULL ABLATION RESULTS

The aggregate ablation comparison in Table 3 is expanded into individual dataset–horizon results below. The controls and their training procedures are defined in Appendix A.4.

Table 9: Detailed ablation results. Ref.: reference model. TF: Teacher-free. Inertia: separately trained Inertia-only model.

		Ref.		DESA		Shuffle		Matched-K		TF		Inertia	
Dataset	H	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.1603	0.2091	0.1629	0.2105	0.1666	0.2143	0.1658	0.2139	0.1655	0.2132	0.1661	0.2136
	192	0.2126	0.2570	0.2144	0.2573	0.2168	0.2582	0.2170	0.2587	0.2159	0.2603	0.2108	0.2525
	336	0.2693	0.2954	0.2707	0.2964	0.2718	0.2957	0.2740	0.2982	0.2730	0.2998	0.2667	0.2923
	720	0.3474	0.3472	0.3485	0.3479	0.3492	0.3477	0.3507	0.3487	0.3507	0.3505	0.3454	0.3428
Solar-Energy	96	0.2041	0.2472	0.2065	0.2555	0.2149	0.2616	0.2453	0.2882	0.2220	0.2565	0.2843	0.3129
	192	0.2404	0.2746	0.2474	0.2870	0.2471	0.2884	0.2865	0.3180	0.2583	0.2875	0.3236	0.3318
	336	0.2542	0.2821	0.2556	0.2892	0.2624	0.3001	0.3121	0.3292	0.2687	0.2958	0.3688	0.3535
	720	0.2456	0.2783	0.2464	0.2795	0.2554	0.2873	0.2455	0.2786	0.2629	0.2945	0.3718	0.3694
Electricity	96	0.1504	0.2476	0.1563	0.2538	0.1568	0.2533	0.1888	0.2869	0.1720	0.2657	0.1783	0.2670
	192	0.1649	0.2590	0.1720	0.2672	0.1708	0.2647	0.1971	0.2934	0.1867	0.2771	0.1881	0.2732
	336	0.1864	0.2783	0.1957	0.2896	0.1942	0.2864	0.2122	0.3055	0.1984	0.2903	0.2037	0.2886
	720	0.2207	0.3103	0.2330	0.3197	0.2348	0.3206	0.2602	0.3443	0.2403	0.3266	0.2447	0.3221
Traffic	96	0.4626	0.3185	0.4788	0.3339	0.4901	0.3466	0.5461	0.3768	0.4740	0.3310	0.6157	0.4006
	192	0.4728	0.3192	0.4882	0.3423	0.4967	0.3475	0.5353	0.3727	0.4776	0.3302	0.5797	0.3738
	336	0.5025	0.3276	0.5041	0.3406	0.5199	0.3541	0.5524	0.3752	0.4936	0.3378	0.5894	0.3752
	720	0.5413	0.3443	0.5495	0.3735	0.5459	0.3645	0.5778	0.3880	0.5334	0.3567	0.6299	0.3939
Exchange	96	0.0868	0.2075	0.0852	0.2047	0.0864	0.2070	0.0963	0.2194	0.1133	0.2354	0.0888	0.2078
	192	0.1765	0.2999	0.1748	0.2981	0.1765	0.2998	0.1860	0.3093	0.2327	0.3435	0.1855	0.3054
	336	0.4241	0.4683	0.3995	0.4541	0.3852	0.4488	0.4036	0.4580	0.4334	0.4720	0.3447	0.4239
	720	0.9043	0.7200	0.8945	0.7155	0.9024	0.7193	0.9381	0.7350	1.0421	0.7584	0.9064	0.7200
ETTh1	96	0.4076	0.4136	0.3974	0.4069	0.4120	0.4162	0.4240	0.4249	0.4393	0.4338	0.3825	0.3919
	192	0.4590	0.4429	0.4567	0.4423	0.4570	0.4400	0.4693	0.4514	0.5281	0.4785	0.4346	0.4216
	336	0.4908	0.4580	0.4924	0.4612	0.4957	0.4608	0.5064	0.4694	0.5779	0.5028	0.4765	0.4413
	720	0.5038	0.4845	0.5023	0.4838	0.5013	0.4838	0.5118	0.4900	0.6431	0.5514	0.4799	0.4620
ETTh2	96	0.2957	0.3462	0.2932	0.3439	0.2982	0.3466	0.3031	0.3515	0.3566	0.3855	0.2950	0.3440
	192	0.3800	0.3950	0.3801	0.3961	0.3795	0.3951	0.3879	0.4013	0.4509	0.4413	0.3853	0.4000
	336	0.4240	0.4334	0.4211	0.4315	0.4208	0.4313	0.4265	0.4354	0.4523	0.4522	0.4299	0.4378
	720	0.4285	0.4427	0.4269	0.4421	0.4270	0.4418	0.4307	0.4451	0.4859	0.4750	0.4469	0.4531
ETTm1	96	0.3422	0.3754	0.3425	0.3755	0.3452	0.3781	0.3663	0.3923	0.3395	0.3727	0.3323	0.3645
	192	0.3829	0.3926	0.3766	0.3888	0.3821	0.3922	0.3907	0.4000	0.3930	0.4013	0.3758	0.3830
	336	0.4060	0.4087	0.4219	0.4190	0.4052	0.4085	0.4221	0.4203	0.4392	0.4299	0.4061	0.4026
	720	0.4921	0.4587	0.4981	0.4616	0.4851	0.4544	0.4906	0.4571	0.5263	0.4752	0.4696	0.4364
ETTm2	96	0.1747	0.2596	0.1829	0.2656	0.1759	0.2596	0.1828	0.2654	0.1943	0.2728	0.1784	0.2597
	192	0.2460	0.3053	0.2453	0.3050	0.2443	0.3038	0.2496	0.3078	0.2819	0.3277	0.2422	0.3017
	336	0.3147	0.3484	0.3162	0.3494	0.3149	0.3479	0.3121	0.3471	0.3552	0.3712	0.3011	0.3398
	720	0.4074	0.4013	0.4067	0.4005	0.4061	0.3998	0.4069	0.4010	0.4641	0.4336	0.3993	0.3995

D SENSITIVITY ANALYSIS

Results for input length and Event width average nine datasets at $H = 96$. Results for the computation penalty weight average all 36 conditions. Each width uses its corresponding reference model. The gamma experiment reuses the same fixed reference models and varies the sparse model's computation penalty weight while retaining the other training settings. Here, $\gamma = 0.1$ is the DESA control, marked by open diamonds in Figure 7.

Table 10: Input length sensitivity ($H = 96$, $d = 64$, $\gamma = 0.1$).

Input length	Active steps (%)	FLOPs (M)	MSE	MAE
96 (DESA)	21.78	44.07	0.2562	0.2945
192	17.18	65.53	0.2405	0.2854
336	13.48	108.56	0.2384	0.2867
720	8.97	281.43	0.2385	0.2912

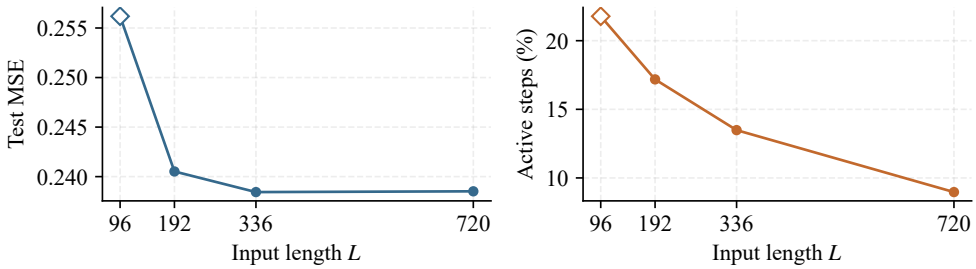


Figure 5: Input length sensitivity at $H = 96$.

Table 11: Event width sensitivity ($L = H = 96$, $\gamma = 0.1$).

Event width	Active steps (%)	FLOPs (M)	MSE	MAE
32	25.51	26.91	0.2585	0.2966
64 (DESA)	21.78	44.07	0.2562	0.2945
128	18.17	76.16	0.2556	0.2941
256	18.50	135.31	0.2536	0.2921
512	20.83	325.57	0.2522	0.2909

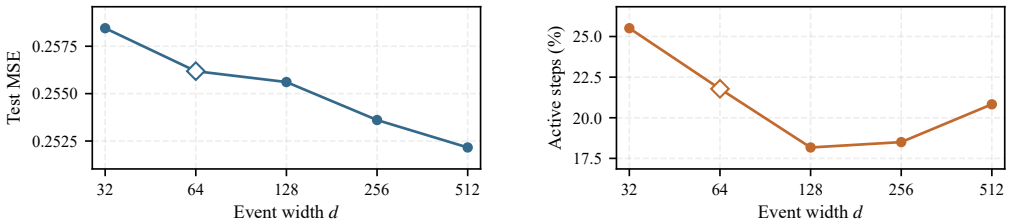


Figure 6: Event width sensitivity at $H = 96$.

Table 12: Gamma sensitivity ($L = 96$, $d = 64$).

Gamma	Active steps (%)	MSE	MAE
0	97.63	0.3439	0.3520
0.01	63.99	0.3436	0.3523
0.1 (DESA)	25.17	0.3457	0.3553
0.3	19.53	0.3462	0.3562
0.5	20.81	0.3468	0.3568

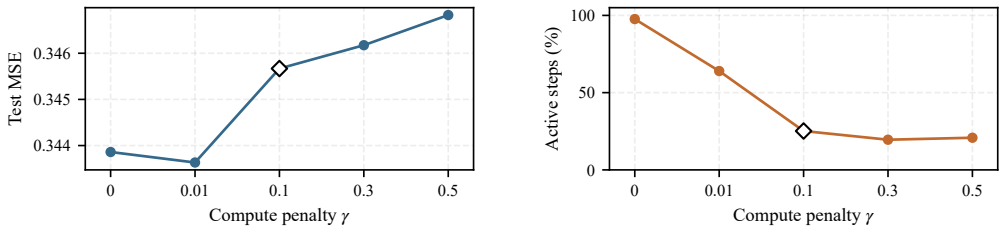


Figure 7: Effect of the computation penalty on error and active-step rate.

Without a computation penalty, the model selects nearly all centers. The penalty encourages time-step selection while allowing a trade-off between forecasting error and active-step rate. Increasing γ from 0.3 to 0.5 does not further reduce the active-step rate, showing that a stronger penalty does not guarantee fewer active centers at the selected checkpoint.

E QUALITATIVE EXAMPLES OF TIME-STEP SELECTION AND FORECAST CORRECTION

Figures 8–16 extend Figure 4 to all nine datasets and four forecast horizons ($H \in \{96, 192, 336, 720\}$, from top to bottom), with $L = 96$. We plot channel 0 of test-window index 240, with the selected count determined by that window. Selection uses information from all variables, although only channel 0 is plotted. The left panels show input histories with selected centers marked in orange. These marks are binary selections, not S-Gate intensities or context windows. The right panels show future ground truth, the base forecast from the same trained model, and the DESA forecast. The shaded gap is the Event correction. Active-step rates and forecast traces describe individual windows, not dataset averages.

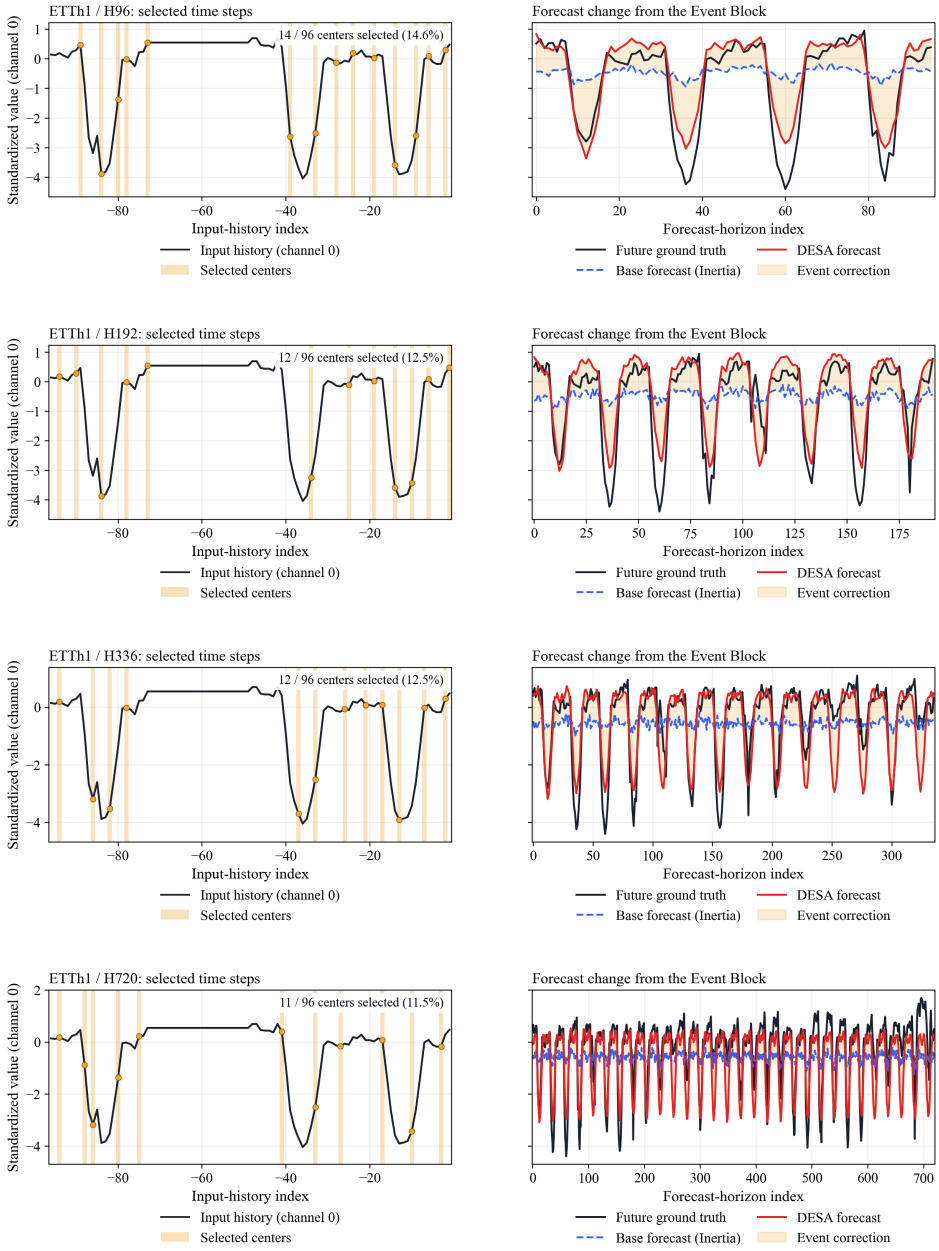


Figure 8: Time-step selection and forecast correction on ETTh1 across four horizons.

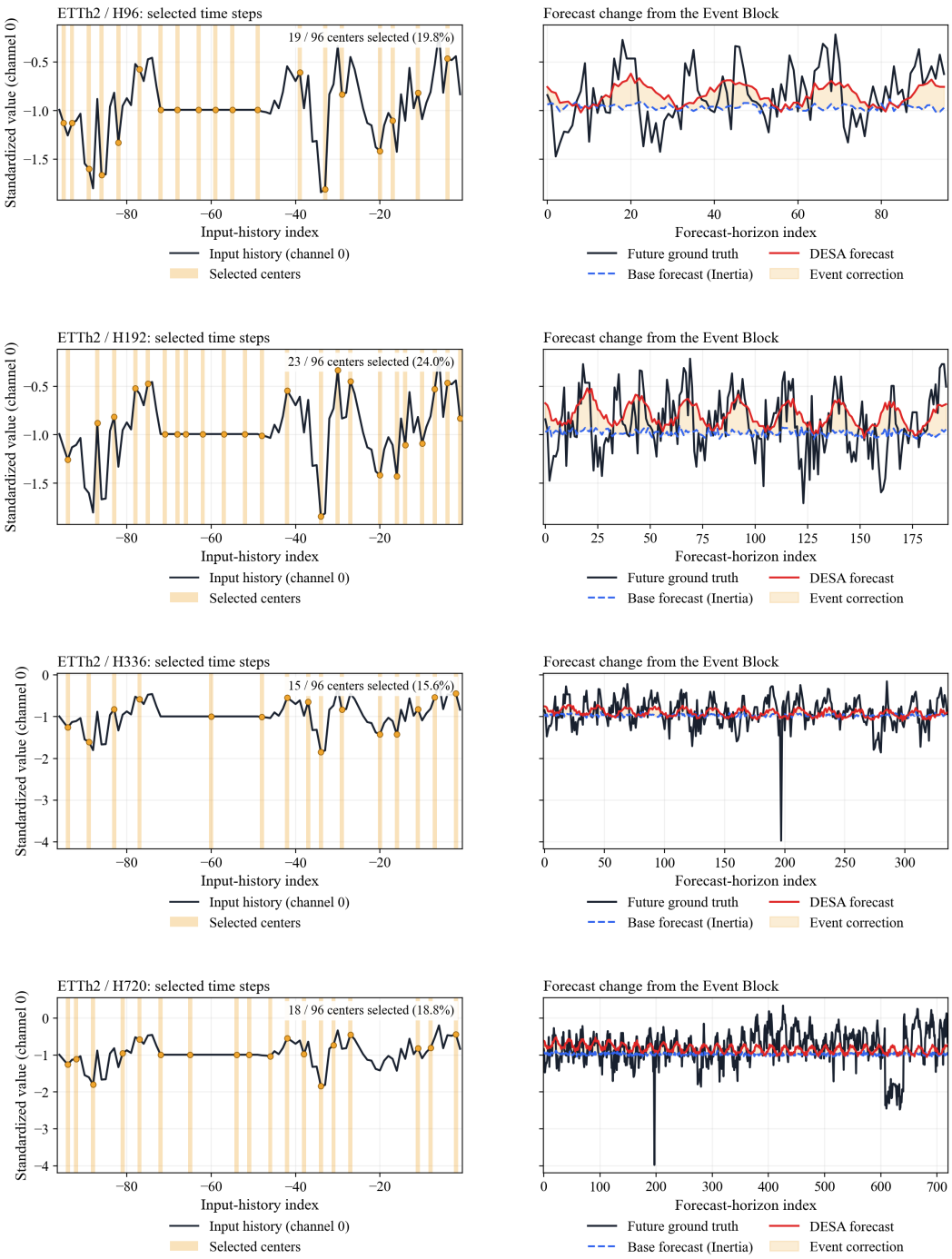


Figure 9: Time-step selection and forecast correction on ETTh2 across four horizons.

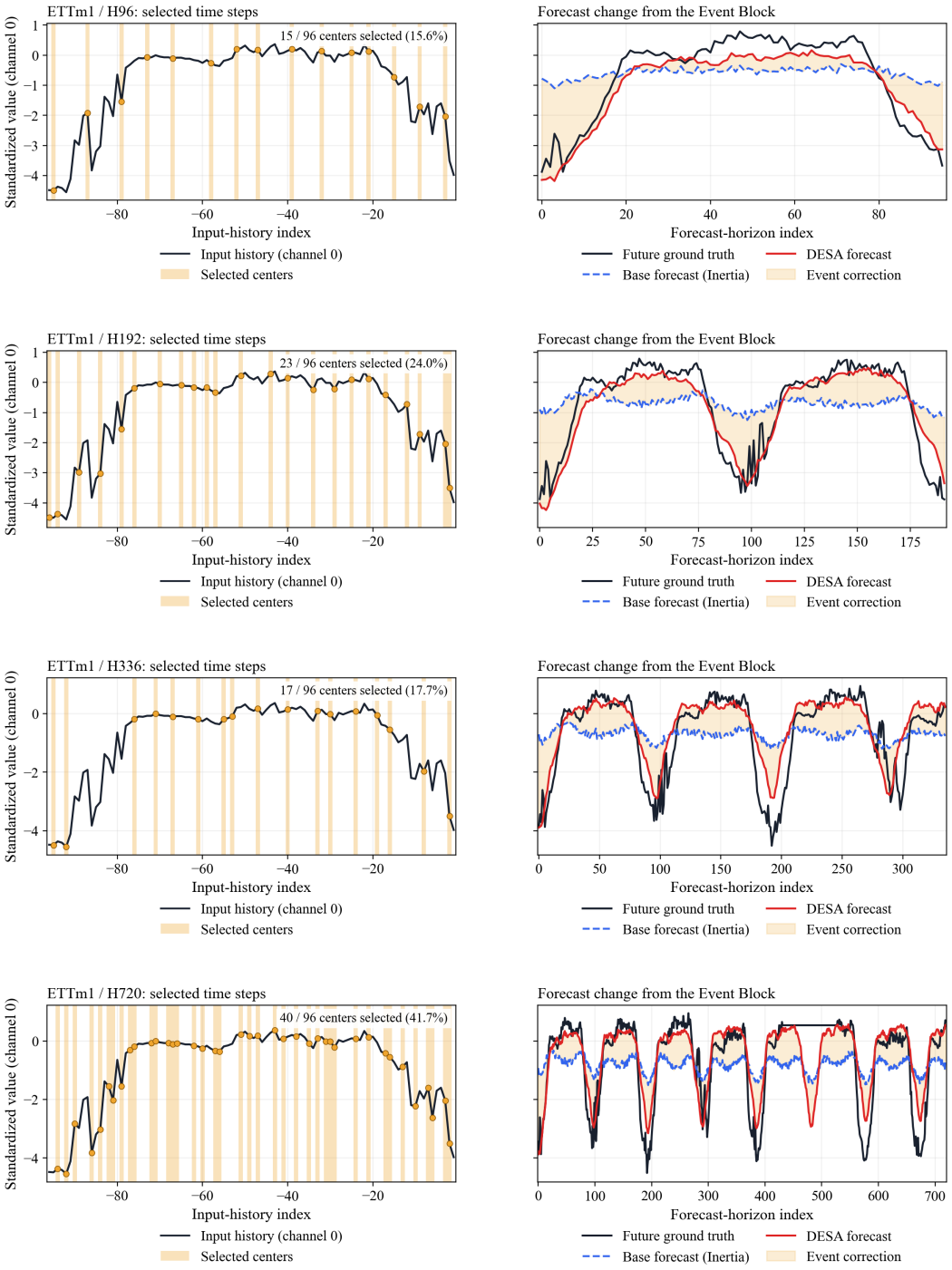


Figure 10: Time-step selection and forecast correction on ETTm1 across four horizons.

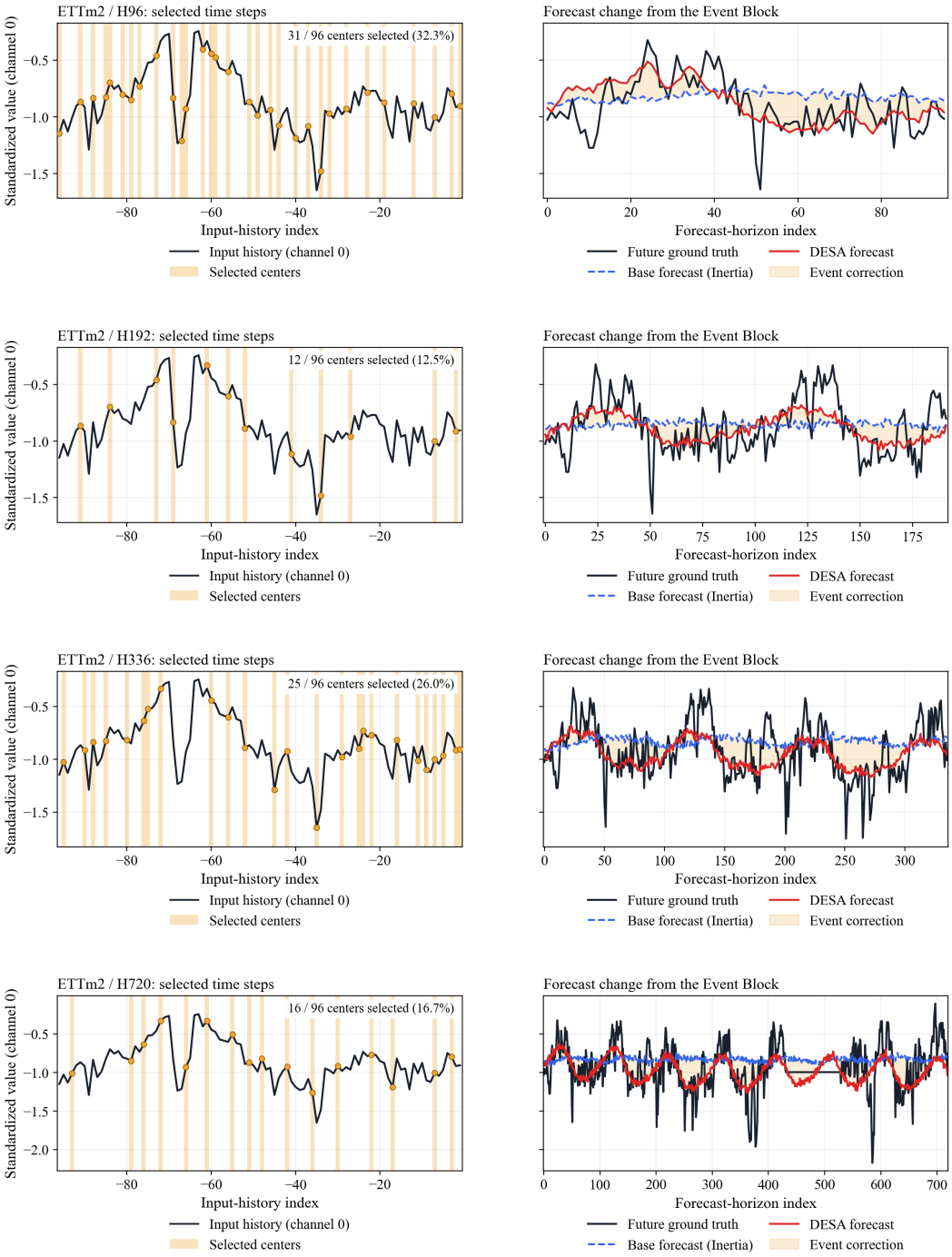


Figure 11: Time-step selection and forecast correction on ETTm2 across four horizons.

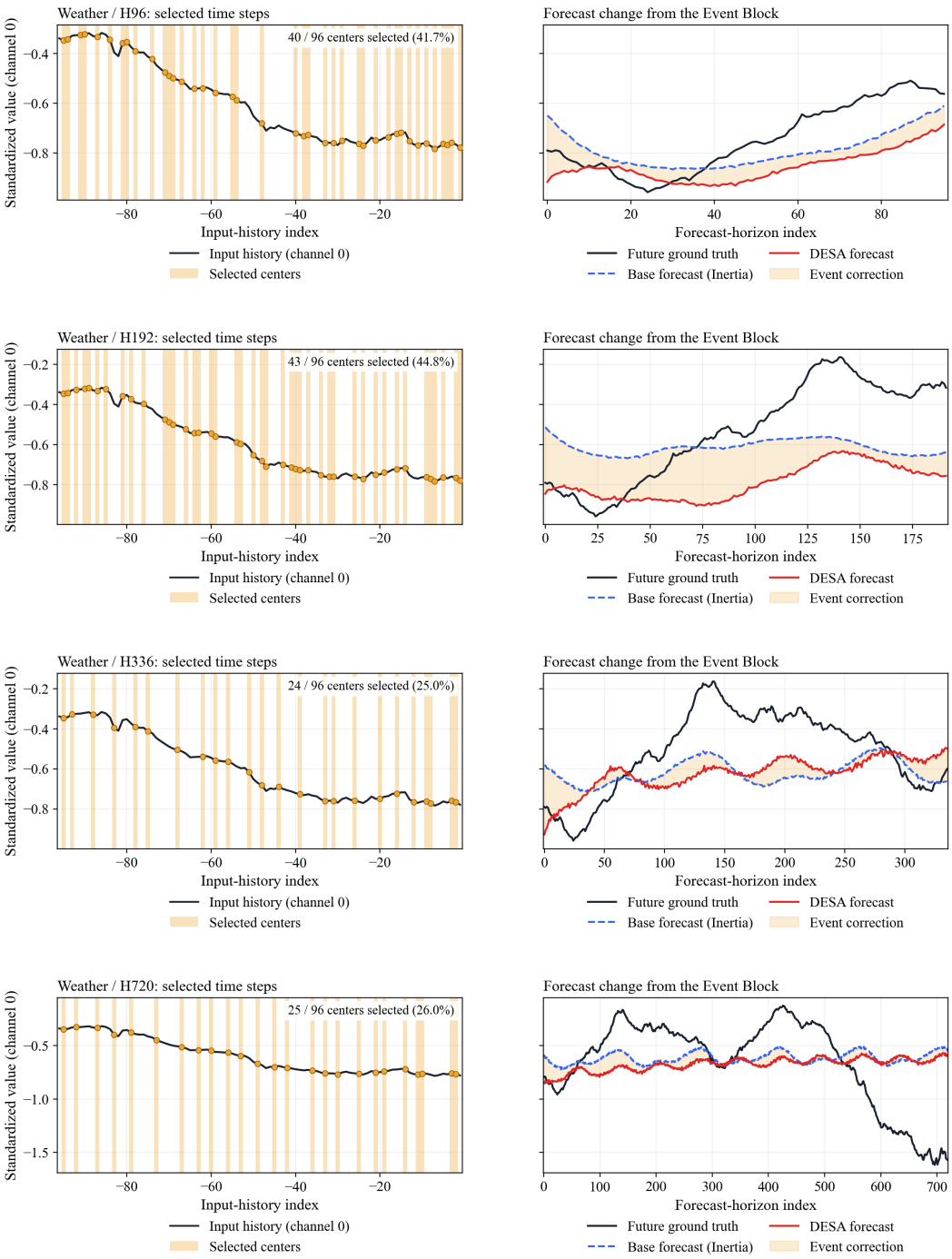


Figure 12: Time-step selection and forecast correction on Weather across four horizons.

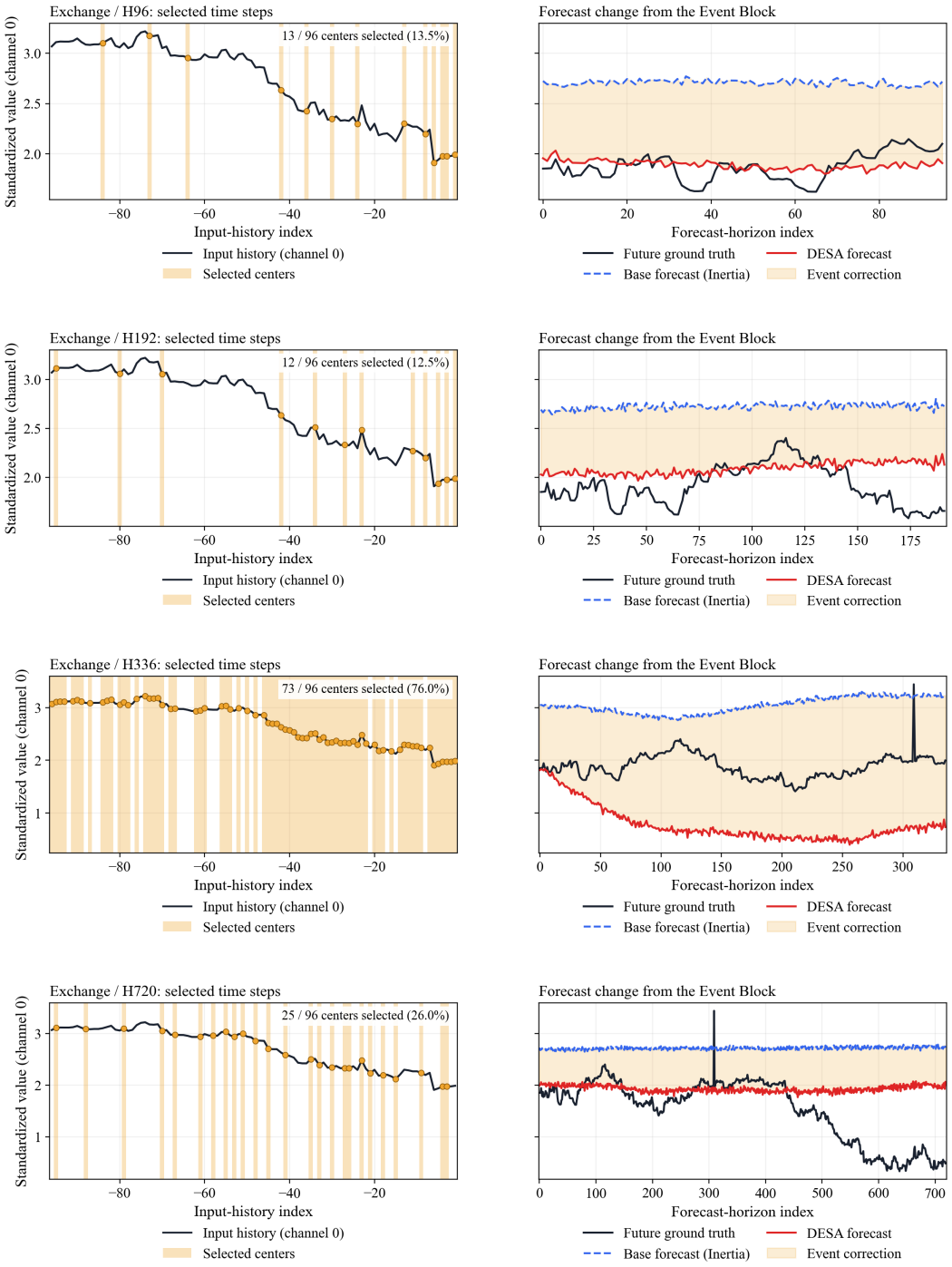


Figure 13: Time-step selection and forecast correction on Exchange across four horizons.

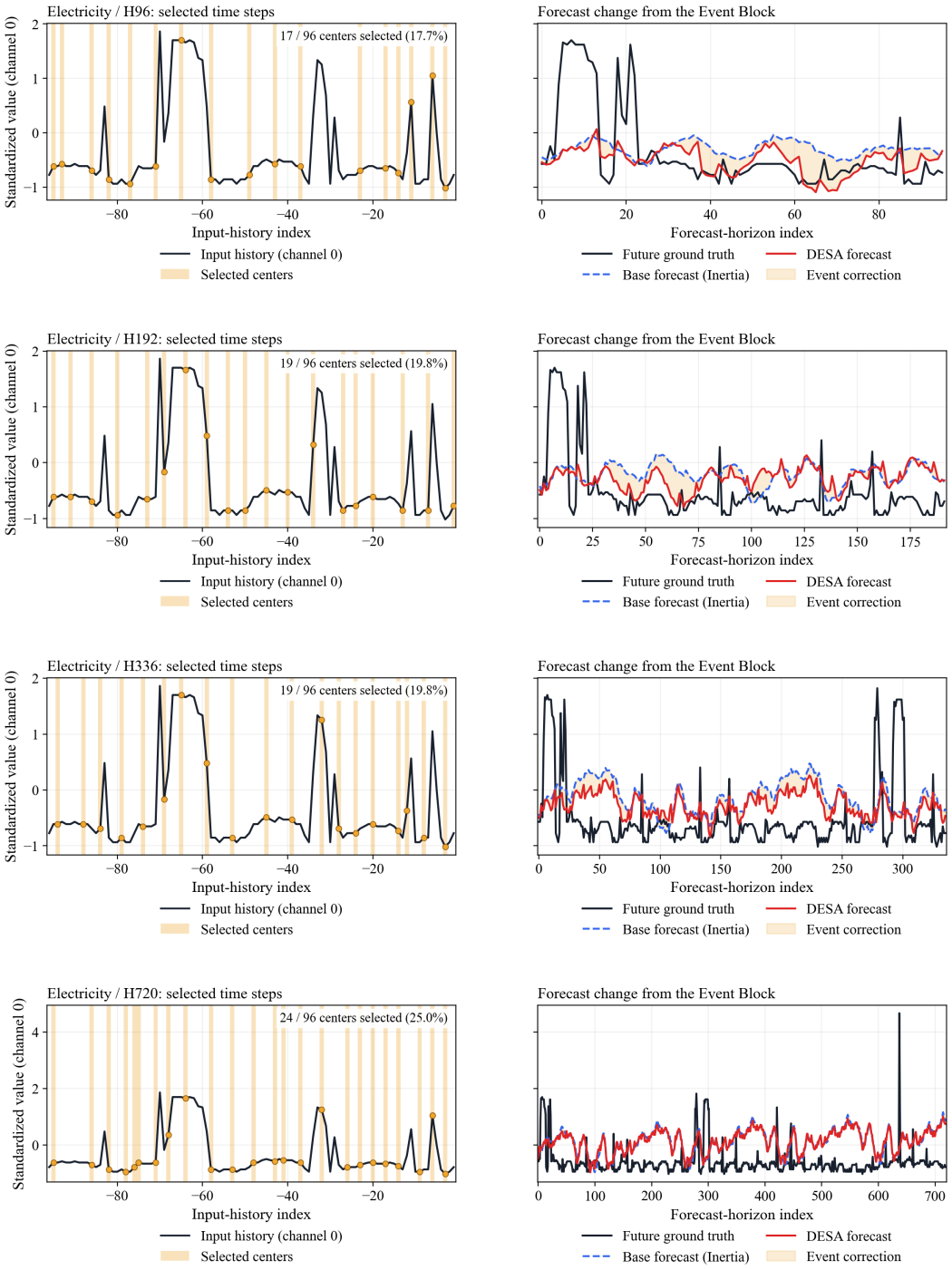


Figure 14: Time-step selection and forecast correction on Electricity across four horizons.

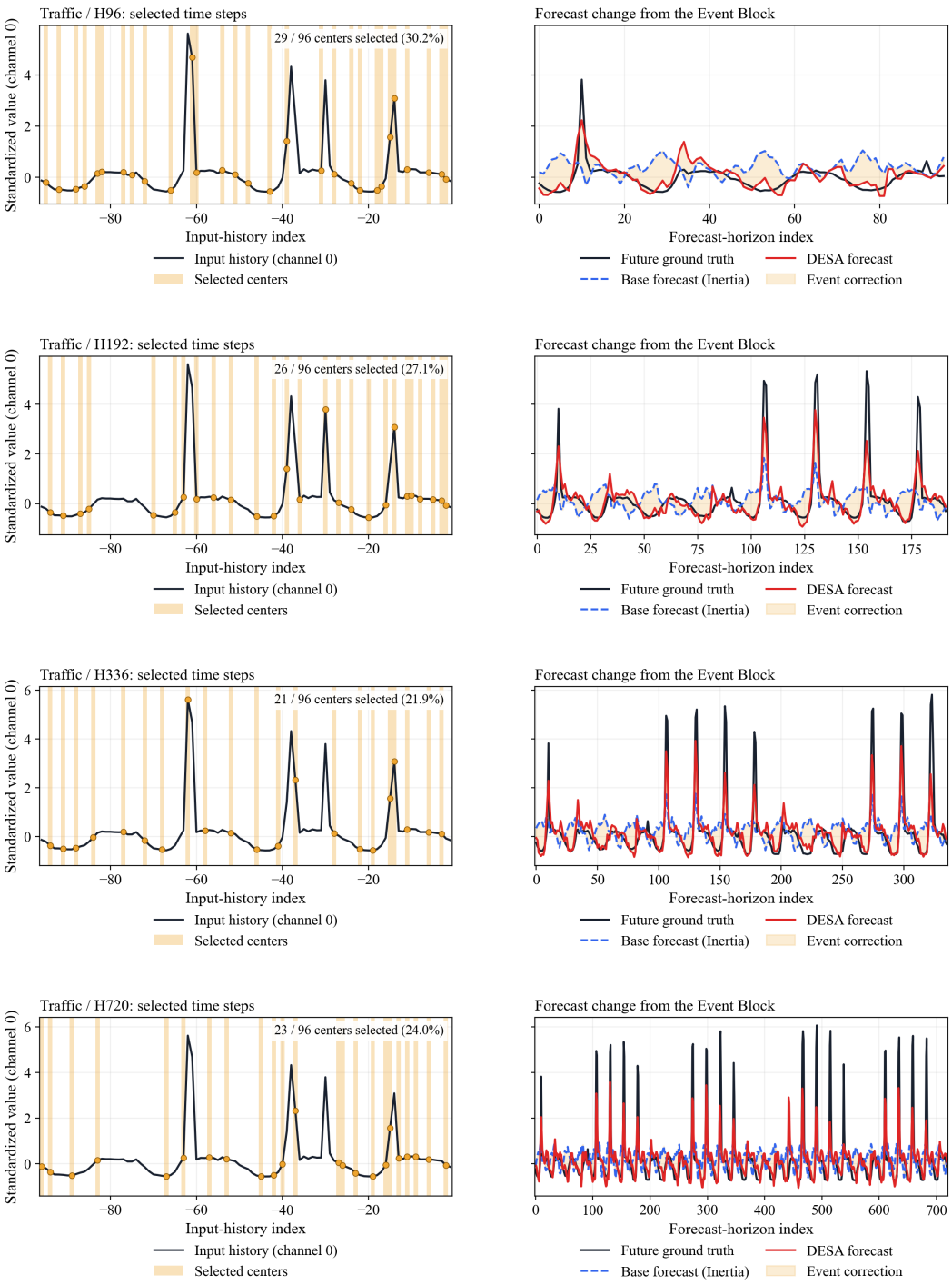


Figure 15: Time-step selection and forecast correction on Traffic across four horizons.

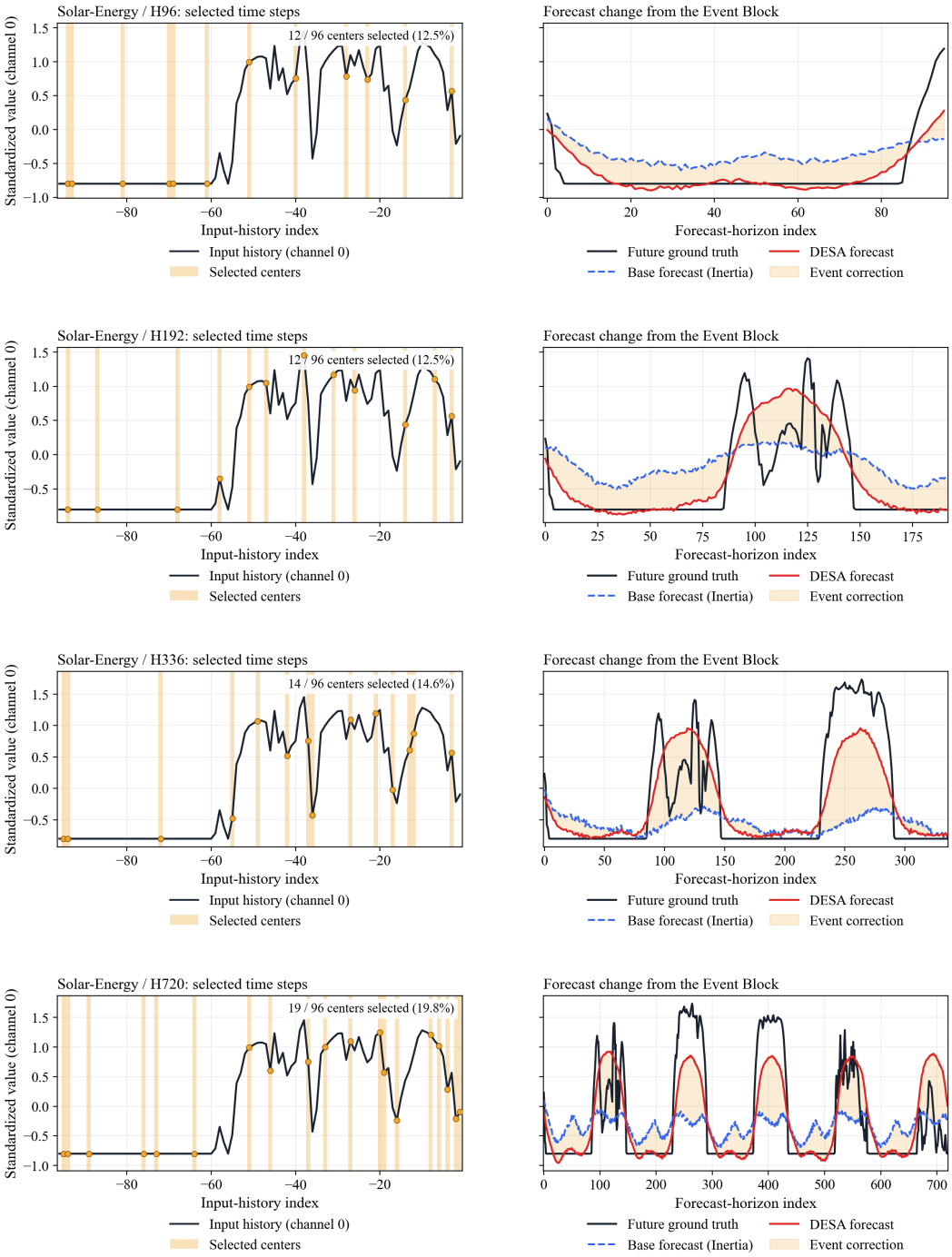


Figure 16: Time-step selection and forecast correction on Solar-Energy across four horizons.

F FULL BENCHMARK RESULTS

Table 13: Full benchmark results ($L = 96$, $H \in \{96, 192, 336, 720\}$). DESA: mean ($\pm$ SD). Avg: four-horizon mean. Colors follow Table 2.

		DESA		Timer-XL		S-Mamba		TimeMixer		iTransformer		PatchTST		Crossformer		TiDE		TimesNet		DLinear	
Dataset	H	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.163** (±0.0003)	0.211 (±0.0002)	0.175	0.217	**0.165**	**0.210**	**0.163**	**0.209**	0.174	0.214	0.186	0.227	0.195	0.271	0.202	0.261	0.172	0.220	0.195	0.252
	192	0.215 (±0.0011)	0.257 (±0.0007)	0.225	0.259	0.214	**0.252**	**0.208**	**0.250**	0.221	0.254	0.234	0.265	**0.209**	0.277	0.242	0.298	0.219	0.261	0.237	0.295
	336	**0.270** (±0.0005)	**0.296** (±0.0003)	0.280	0.298	0.274	0.297	**0.251**	**0.287**	0.278	**0.296**	0.284	0.301	0.273	0.332	0.287	0.335	0.280	0.306	0.282	0.331
	720	0.348 (±0.0005)	0.348 (±0.0004)	0.359	0.349	0.350	**0.345**	**0.339**	**0.341**	0.358	0.347	0.356	0.349	0.379	0.401	0.351	0.386	0.365	0.359	**0.345**	0.382
	Avg	**0.249** (±0.0002)	0.278 (±0.0001)	0.260	0.281	0.251	**0.276**	**0.240**	**0.272**	0.258	0.278	0.265	0.286	0.264	0.320	0.271	0.320	0.259	0.287	0.265	0.315
Solar-Energy	96	0.212 (±0.0079)	0.257 (±0.0042)	0.224	0.260	0.205	**0.244**	**0.189**	0.259	**0.203**	**0.237**	0.265	0.323	0.232	0.302	0.312	0.399	0.373	0.358	0.290	0.378
	192	0.245 (±0.0025)	0.284 (±0.0030)	0.275	0.291	0.237	**0.270**	**0.222**	0.283	**0.233**	**0.261**	0.288	0.332	0.371	0.410	0.339	0.416	0.397	0.376	0.320	0.398
	336	0.261 (±0.0055)	0.293 (±0.0036)	0.318	0.316	0.258	**0.288**	**0.231**	0.292	**0.248**	**0.273**	0.301	0.339	0.495	0.515	0.368	0.430	0.420	0.380	0.353	0.415
	720	0.253 (±0.0059)	**0.285** (±0.0050)	0.384	0.345	0.260	0.288	**0.223**	**0.285**	**0.249**	**0.275**	0.295	0.336	0.526	0.542	0.370	0.425	0.420	0.381	0.357	0.413
	Avg	0.243 (±0.0037)	0.280 (±0.0020)	0.300	0.303	0.240	**0.273**	**0.216**	0.280	**0.233**	**0.262**	0.287	0.333	0.406	0.442	0.347	0.418	0.403	0.374	0.330	0.401
Electricity	96	0.156 (±0.0005)	0.253 (±0.0006)	**0.135**	**0.229**	**0.139**	**0.235**	0.153	0.247	0.148	0.240	0.190	0.296	0.219	0.314	0.237	0.329	0.168	0.272	0.210	0.302
	192	0.171 (±0.0018)	0.265 (±0.0024)	**0.156**	**0.250**	**0.159**	0.255	0.166	0.256	0.162	**0.253**	0.199	0.304	0.231	0.322	0.236	0.330	0.184	0.289	0.210	0.305
	336	0.193 (±0.0023)	0.286 (±0.0033)	**0.176**	**0.272**	**0.176**	**0.272**	0.185	0.277	**0.178**	**0.269**	0.217	0.319	0.246	0.337	0.249	0.344	0.198	0.300	0.223	0.319
	720	0.231 (±0.0019)	0.319 (±0.0017)	0.227	0.317	**0.204**	**0.298**	0.225	**0.310**	0.225	0.317	0.258	0.352	0.280	0.363	0.284	0.373	**0.220**	0.320	0.258	0.350
	Avg	0.188 (±0.0015)	0.281 (±0.0019)	**0.174**	**0.267**	**0.170**	**0.265**	0.182	0.273	0.178	0.270	0.216	0.318	0.244	0.334	0.252	0.344	0.193	0.295	0.225	0.319
Traffic	96	0.481 (±0.0023)	0.337 (±0.0033)	0.473	0.319	**0.382**	**0.261**	0.462	0.285	**0.395**	**0.268**	0.526	0.347	0.644	0.429	0.805	0.493	0.593	0.321	0.650	0.396
	192	0.486 (±0.0021)	0.340 (±0.0025)	0.496	0.330	**0.396**	**0.267**	0.473	0.296	**0.417**	**0.276**	0.522	0.332	0.665	0.431	0.756	0.474	0.617	0.336	0.598	0.370
	336	0.502 (±0.0017)	0.339 (±0.0021)	0.539	0.346	**0.417**	**0.276**	0.498	0.296	**0.433**	**0.283**	0.517	0.334	0.674	0.420	0.762	0.477	0.629	0.336	0.605	0.373
	720	0.546 (±0.0080)	0.369 (±0.0086)	0.655	0.388	**0.460**	**0.300**	0.506	0.313	**0.467**	**0.302**	0.552	0.352	0.683	0.424	0.719	0.449	0.640	0.350	0.645	0.394
	Avg	0.504 (±0.0019)	0.346 (±0.0023)	0.541	0.346	**0.414**	**0.276**	0.485	0.298	**0.428**	**0.282**	0.529	0.341	0.667	0.426	0.761	0.473	0.620	0.336	0.625	0.383
Exchange	96	**0.086** (±0.0006)	**0.206** (±0.0010)	**0.086**	**0.206**	**0.086**	0.207	**0.086**	**0.205**	**0.086**	**0.206**	**0.088**	**0.205**	0.256	0.367	0.094	0.218	0.107	0.234	**0.088**	0.218
	192	**0.174** (±0.0009)	**0.298** (±0.0008)	0.177	0.300	0.182	0.304	0.193	0.312	0.177	**0.299**	**0.176**	**0.299**	0.470	0.509	0.184	0.307	0.226	0.344	**0.176**	0.315
	336	0.397 (±0.0021)	0.453 (±0.0013)	0.318	**0.410**	0.332	0.418	0.356	0.433	0.331	0.417	**0.301**	**0.397**	1.268	0.883	0.349	0.431	0.367	0.448	**0.313**	0.427
	720	0.891 (±0.0046)	0.713 (±0.0024)	**0.836**	**0.690**	0.867	0.703	0.912	0.712	0.847	**0.691**	0.901	0.714	1.767	1.068	0.852	0.698	0.964	0.746	**0.839**	0.695
	Avg	0.387 (±0.0016)	0.417 (±0.0007)	**0.354**	**0.402**	0.367	0.408	0.387	0.416	**0.360**	**0.403**	0.367	0.404	0.940	0.707	0.370	0.414	0.416	0.443	**0.354**	0.414
ETTh1	96	0.404 (±0.0057)	0.412 (±0.0044)	**0.379**	**0.397**	0.386	0.405	**0.375**	**0.400**	0.386	0.405	0.460	0.447	0.423	0.448	0.479	0.464	0.384	0.402	0.397	0.412
	192	0.462 (±0.0059)	0.446 (±0.0045)	**0.429**	**0.428**	0.443	0.437	**0.429**	**0.421**	0.441	0.436	0.460	0.445	0.471	0.474	0.525	0.492	**0.436**	0.429	0.446	0.441
	336	0.496 (±0.0093)	0.465 (±0.0085)	**0.469**	**0.451**	0.489	0.468	**0.484**	**0.458**	0.487	**0.458**	0.546	0.496	0.570	0.546	0.565	0.515	0.491	0.469	0.489	0.467
	720	0.502 (±0.0005)	0.485 (±0.0011)	**0.492**	**0.476**	0.502	0.489	**0.498**	**0.482**	0.503	0.491	0.544	0.517	0.653	0.621	0.594	0.558	0.521	0.500	0.513	0.510
	Avg	0.466 (±0.0048)	0.452 (±0.0042)	**0.442**	**0.438**	0.455	0.450	**0.447**	**0.440**	0.454	0.448	0.503	0.476	0.529	0.522	0.541	0.507	0.458	0.450	0.461	0.458
ETTh2	96	**0.295** (±0.0014)	**0.345** (±0.0010)	0.302	0.349	0.296	0.348	**0.289**	**0.341**	0.297	0.349	0.308	0.355	0.745	0.584	0.400	0.440	0.340	0.374	0.340	0.394
	192	0.378 (±0.0020)	**0.395** (±0.0015)	0.385	0.400	**0.376**	0.396	**0.372**	**0.392**	0.380	0.400	0.393	0.405	0.877	0.656	0.528	0.509	0.402	0.414	0.482	0.479
	336	**0.423** (±0.0023)	0.433 (±0.0018)	0.439	0.439	0.424	**0.431**	**0.386**	**0.414**	0.428	0.432	0.427	0.436	1.043	0.731	0.643	0.571	0.452	0.452	0.591	0.541
	720	0.428 (±0.0008)	**0.443** (±0.0008)	0.459	0.461	**0.426**	0.444	**0.412**	**0.434**	0.427	0.445	0.436	0.450	1.104	0.763	0.874	0.679	0.462	0.468	0.839	0.661
	Avg	**0.381** (±0.0011)	**0.404** (±0.0007)	0.397	0.412	**0.381**	0.405	**0.365**	**0.395**	0.383	0.407	0.391	0.412	0.942	0.684	0.611	0.550	0.414	0.427	0.563	0.519
ETTm1	96	0.341 (±0.0017)	0.374 (±0.0015)	**0.327**	**0.365**	0.333	0.368	**0.320**	**0.357**	0.334	0.368	0.352	0.374	0.404	0.426	0.364	0.387	0.338	0.375	0.346	0.374
	192	0.385 (±0.0095)	0.395 (±0.0060)	0.393	0.404	0.376	0.390	**0.361**	**0.381**	0.377	0.391	**0.367**	**0.385**	0.450	0.451	0.398	0.404	0.374	0.387	0.382	0.391
	336	0.417 (±0.0117)	0.416 (±0.0075)	0.459	0.442	**0.408**	0.413	**0.390**	**0.404**	0.426	0.420	0.421	0.414	0.532	0.515	0.428	0.425	0.410	**0.411**	0.415	0.415
	720	0.495 (±0.0035)	0.459 (±0.0022)	0.582	0.502	0.475	**0.448**	**0.454**	**0.441**	0.491	0.459	**0.462**	0.449	0.666	0.589	0.487	0.461	0.478	0.450	0.473	0.451
	Avg	0.409 (±0.0019)	0.411 (±0.0013)	0.440	0.428	**0.398**	**0.405**	**0.381**	**0.396**	0.407	0.410	0.401	0.406	0.513	0.495	0.419	0.419	0.400	0.406	0.404	0.408
ETTm2	96	**0.178** (±0.0041)	**0.262** (±0.0032)	0.179	**0.262**	0.179	0.263	**0.175**	**0.258**	0.180	0.264	0.183	0.270	0.287	0.366	0.207	0.305	0.187	0.267	0.193	0.293
	192	0.247 (±0.0016)	0.307 (±0.0016)	**0.246**	**0.306**	0.250	0.309	**0.237**	**0.299**	0.250	0.309	0.255	0.314	0.414	0.492	0.290	0.364	0.249	0.309	0.284	0.361
	336	0.314 (±0.0020)	0.348 (±0.0013)	0.315	0.350	0.312	0.349	**0.298**	**0.340**	0.311	0.348	**0.309**	**0.347**	0.597	0.542	0.377	0.422	0.321	0.351	0.382	0.429
	720	**0.406** (±0.0012)	**0.400** (±0.0007)	0.425	0.413	0.411	0.406	**0.391**	**0.396**	0.412	0.407	0.412	0.404	1.730	1.042	0.558	0.524	0.408	0.403	0.558	0.525
	Avg	**0.286** (±0.0011)	**0.329** (±0.0007)	0.291	0.333	0.288	0.332	**0.275**	**0.323**	0.288	0.332	0.290	0.334	0.757	0.611	0.358	0.404	0.291	0.333	0.354	0.402
Overall		0.346 (±0.0004)	0.355 (±0.0002)	0.355	0.357	**0.329**	**0.343**	**0.331**	**0.344**	0.332	**0.343**	0.361	0.368	0.585	0.505	0.437	0.428	0.384	0.372	0.398	0.402