

---

# Native Sparse Time Attention for High-Dimensional Multivariate Forecasting

---

**Balthazar Courvoisier**

LAMSADE, Université Paris Dauphine – PSL  
Queensfield AI Technologies  
Paris, France  
balthazar.courvoisier@dauphine.psl.eu

**Tristan Cazenave**

LAMSADE, Université Paris Dauphine – PSL  
Paris, France

## Abstract

Scaling attention-based forecasting models to high-dimensional multivariate time series often comes at the cost of cross-channel interactions. Many existing efficient architectures avoid the quadratic cost of dense attention by restricting communication to compressed representations or removing multivariate interactions altogether. We introduce Native Sparse Time Attention, which enables fine-grained interactions while scaling linearly in the number of channels for fixed sparsity budgets. This efficiency exposes a tunable accuracy–compute frontier and enables longer contexts, which can improve forecasting performance under memory constraints.

## 1 Introduction

Patch-based multivariate Transformers can preserve both when an observation occurred and which channel produced it, but dense attention over the resulting grid is expensive. With  $P$  patches and  $C$  channels, the grid requires up to  $P^2C^2$  attention scores per head. This limitation becomes a bottleneck on the GTWSF dataset, where seven patches and 3,850 channels produce 26,950 tokens. Dense attention over this grid evaluates up to 726 million attention scores per head in every layer. Such quadratic growth makes dense cross-channel attention particularly costly in the high-dimensional settings where cross-channel information may be most useful (Liu et al., 2025).

Eliminating cross-channel interactions entirely avoids this cost, but also removes potentially useful dependencies among series. Yet channel-independent models remain competitive with elaborate multivariate architectures (Zeng et al., 2023; Nie et al., 2023; Xu et al., 2024), suggesting that exhaustive interaction between all pairs of channels is often unnecessary. Several multivariate architectures therefore avoid dense pairwise attention while retaining some cross-channel interaction, including approaches based on routing mechanisms (Zhang & Yan, 2023; Liu et al., 2024a). A remaining design challenge is to preserve the efficiency gains of compression while retaining direct access to the original tokens when cross-channel interactions are needed.

Our key idea is to make the relationship type explicit in the attention operator. We decompose the visible history into three complementary views (cf. Figure 2). A *univariate branch* attends densely to the query channel’s visible history, a *compression branch* compresses each cross-section into a system-level state, and a *selection branch* retrieves a small set of patches and channels at full resolution. This decomposition preserves exact same-channel temporal dependence while representing cross-channel context through compressed and selectively retrieved interactions rather than exhaustive pairwise attention. The three-path design is inspired by Native Sparse Attention (Yuan et al., 2025), but

reinterprets its local, compression, and selection paths according to the structure of multivariate time series.

Our main contributions are:

- We introduce *Native Sparse Time Attention* (NSTA), a three-path attention operator that distinguishes same-channel and cross-channel relationships without channel positional embeddings or channel-indexed parameters (Section 3).
- We show that NSTA’s attention computation and score-activation memory scale linearly with the number of channels for fixed temporal and channel-selection budgets (Section 3.6).
- We show that NSTA exposes a controllable accuracy–compute frontier relative to dense Time Attention. Sparse operating points can match or modestly improve dense attention at substantially lower attention cost on some datasets, whereas stronger sparsification incurs a controlled accuracy cost on others (Section 4.3).

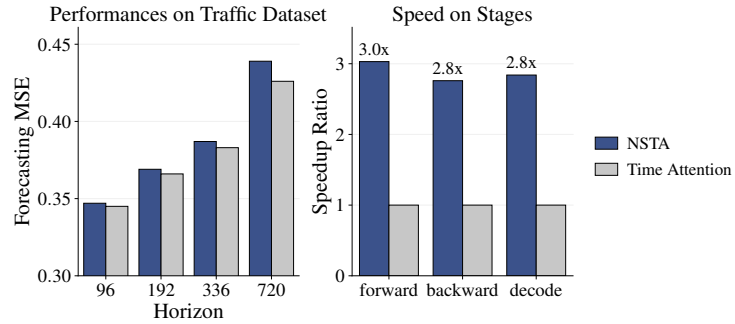


Figure 1: Left: forecasting MSE on Traffic ( $C = 862$ ) at fixed budgets  $k_p = 4$ ,  $k_c = 32$ . Right: speedup by stage at  $C = 862$ . Sparsification costs little accuracy while reducing wall-clock cost.

We discuss related approaches and their relationship to Native Sparse Time Attention in Section A.

## 2 Rethinking Relationship Modelling

Multivariate time series exhibit different types of dependencies. At the univariate level, the future evolution of each variable may depend on specific past observations through temporal structures such as autocorrelation or seasonality (Box et al., 2015). Capturing these dependencies naturally requires direct access to individual historical observations.

Cross-channel dependencies can take more heterogeneous forms. In particular, multiple variables may co-move in response to common underlying factors (Stock & Watson, 2002; Fama & French, 1993). When such shared structure is present, a substantial part of the relevant information can often be summarized through a compact representation of the overall cross-sectional state. Other dependencies are more localized. Idiosyncratic shocks may propagate across a limited set of related channels, much like congestion spreading to neighboring links in a traffic network or lead-lag relationships in finance (Zhao & Shen, 2024; Hou, 2007). Capturing such interactions requires access to the relevant channel-specific observations, as they cannot, in general, be recovered from an aggregate summary alone. We therefore use different attention operators for different dependency types.

Table 1: Cross-channel attention design space.

| Cross-channel strategy | Fine-grained access | Channel complexity |
|------------------------|---------------------|--------------------|
| Independent            | ✗                   | $\mathcal{O}(C)$   |
| Compressed             | ✗                   | $\mathcal{O}(C)$   |
| Dense                  | ✓                   | $\mathcal{O}(C^2)$ |
| <b>Sparse</b>          | ✓                   | $\mathcal{O}(C)$   |

We call a cross-channel interaction *fine-grained* when a query can attend directly to individual channel-specific representations rather than only to a pooled or intermediate summary. Dense attention over the full grid provides this access, but requires up to  $\mathcal{O}(P^2C^2)$  attention scores.

Most existing efficient designs reduce this cost by restricting cross-channel interactions. Channel-independent models process each channel separately (Zeng et al., 2023; Nie et al., 2023; Xu et al., 2024), yielding linear complexity in  $C$  but eliminating direct cross-channel attention. Other approaches mediate channel interactions through compressed or intermediate representations (Zhang & Yan, 2023; Liu et al., 2024a), reducing computation at the cost of direct fine-grained access to individual channel representations.

### 3 Architecture

#### 3.1 Overview

Native Sparse Time Attention operates through three branches, each defining a different query-key-value representation and attention pattern. Following Liu et al. (2025), RoPE is applied along the temporal axis in each attention branch. The outputs of the three branches are combined through learned gates, as described in Section 3.5.

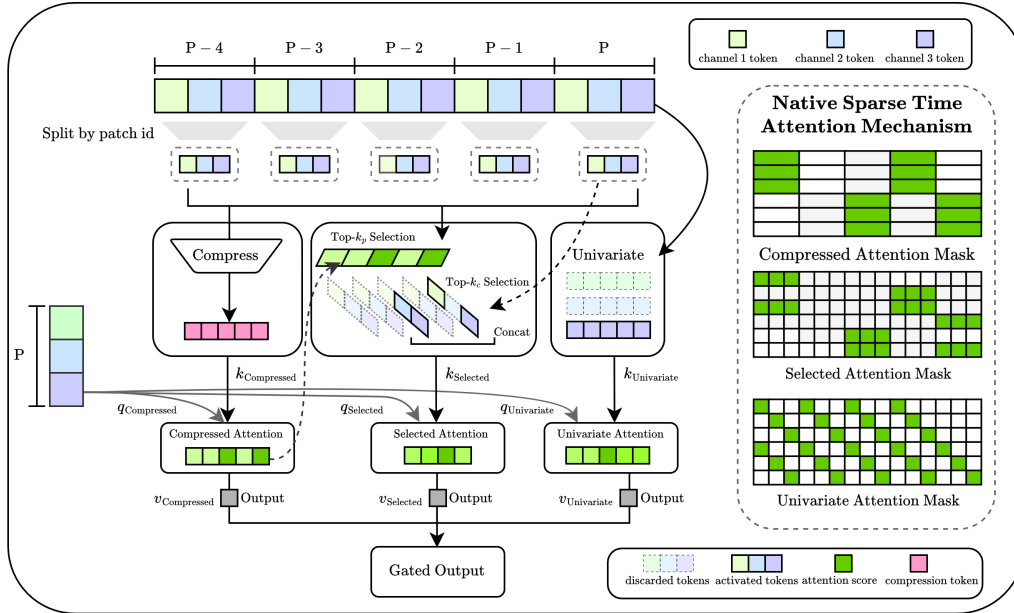


Figure 2: Overview of the Native Sparse Time Attention mechanism, visually inspired by Yuan et al. (2025). Left: the three parallel attention branches. Right: Corresponding attention patterns, with green indicating computed attention scores and white indicating skipped interactions.

The mechanism supports both causal and non-causal attention by applying the corresponding visibility constraints consistently across all three branches. The mechanism is applied independently across attention heads. For clarity, we describe a single head below and omit the head index from the notation.

#### 3.2 Univariate Branch

Let  $\mathbf{h}_{p,c} \in \mathbb{R}^D$  denote the embedded token at patch  $p$  and channel  $c$ . For each channel independently, the univariate branch applies standard self-attention over the sequence  $(\mathbf{h}_{p,c})_{p=1}^P$  along the patch axis. Queries attend only to observations from the same channel, preserving exact temporal access at full resolution without introducing cross-channel interactions; the channel axis contributes only a linear factor to the cost, since no query attends across channels. This branch supplies the purely temporal component of the representation, while cross-channel structure is provided by the compression and selection branches (Sections 3.3 and 3.4).

Each channel token is projected with branch-specific matrices  $\mathbf{W}_Q^u, \mathbf{W}_K^u, \mathbf{W}_V^u \in \mathbb{R}^{d_{\text{model}} \times D}$  into univariate queries, keys, and values. Each query attends over the visible patches of its own channel,  $\mathcal{P}_p = \{1, \dots, p\}$ .

### 3.3 Compression Branch

At each patch, the compression branch summarizes the  $C$  channel representations into a single system-level token. This branch aims to capture broad system-level context, while fine-grained cross-channel interactions are delegated to the selection branch (Section 3.4).

Let  $\mathbf{H}_p = [\mathbf{h}_{p,1}, \dots, \mathbf{h}_{p,C}]^\top \in \mathbb{R}^{C \times D}$  collect the channel representations at patch  $p$ . A learned scorer  $\mathbf{u}_{\text{pool}} \in \mathbb{R}^D$  assigns pooling weights  $\mathbf{w}_p$  over channels, which are used to form the compressed representation  $\bar{\mathbf{h}}_p \in \mathbb{R}^D$ :

$$\mathbf{w}_p = \text{softmax}\left(\frac{\mathbf{H}_p \mathbf{u}_{\text{pool}}}{\sqrt{D}}\right), \quad \bar{\mathbf{h}}_p = \mathbf{H}_p^\top \mathbf{w}_p = \sum_{c=1}^C w_{p,c} \mathbf{h}_{p,c} \quad (1)$$

The pooled representation is projected with branch-specific matrices  $\mathbf{W}_K^c, \mathbf{W}_V^c \in \mathbb{R}^{d_{\text{model}} \times D}$  into compression keys and values, while channel tokens are projected into branch-specific queries with  $\mathbf{W}_Q^c \in \mathbb{R}^{d_{\text{model}} \times D}$ . Each query then applies standard attention over the compressed representations of the visible patches  $\mathcal{P}_p = \{1, \dots, p\}$ .

$$\alpha_{p,c}^c = \text{softmax}_{p' \in \mathcal{P}_p} \left( \frac{\langle \mathbf{q}_{p',c}^c, \mathbf{k}_{p'}^c \rangle}{\sqrt{d_{\text{model}}}} \right) \quad (2)$$

### 3.4 Selection Branch

The selection branch performs hierarchical sparse retrieval across both time and channels. It reuses quantities produced by the compression branch: the compressed representation  $\bar{\mathbf{h}}_r$  for patch-level routing and the pooling weights  $w_{r,c}$  (Eq. 1) for channel-level routing. As illustrated in Figure 3, it first selects  $k_p$  historical patches and then  $k_c$  channels within each retained patch.

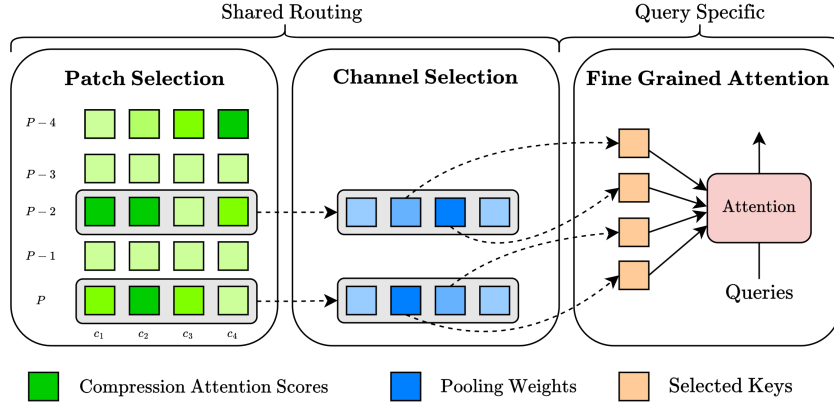


Figure 3: Overview of the hierarchical patch and channel selection mechanism.

**Patch selection.** For query patch  $p$ , we rank visible historical patches by the compression attention score (Eq. 2) averaged across channel queries:

$$s_p = \frac{1}{C} \sum_{c=1}^C \alpha_{p,c}^c, \quad \mathcal{R}_p = \text{top-k}_{p' \in \mathcal{P}_p} (s_{p,p'}; k_p). \quad (3)$$

**Channel selection.** For each retained patch  $r \in \mathcal{R}_p$ , the  $k_c$  channels with largest pooling weights are retained, yielding at most  $k_p k_c$  fine-grained candidates:

$$\mathcal{C}_r = \text{top-k}_{c \in \{1, \dots, C\}} (w_{r,c}; k_c), \quad \mathcal{S}_p = \{(r, c) : r \in \mathcal{R}_p, c \in \mathcal{C}_r\}. \quad (4)$$

Both routing stages are shared across channel queries:  $\mathcal{R}^p$  is common to all queries in patch  $p$ , while  $\mathcal{C}_r$  depends only on the retained patch. This query-shared routing is what preserves linear scaling in  $C$ . Although routing is shared, each query computes its own attention weights over the selected candidates.

The selected original tokens  $\mathbf{h}_{r,c}$  are projected with branch-specific matrices  $\mathbf{W}_Q^s, \mathbf{W}_K^s, \mathbf{W}_V^s \in \mathbb{R}^{d_{\text{model}} \times D}$ . Each channel query attends over  $\mathcal{S}_p$ , with logits biased by the pooling weight of each retained token:

$$\alpha_{p,c}^s = \text{softmax}_{(r,c') \in \mathcal{S}_p} \left( \frac{\langle \mathbf{q}_{p,c}^s, \mathbf{k}_{r,c'}^s \rangle}{\sqrt{d_{\text{model}}}} + \lambda \log w_{r,c'} \right) \quad (5)$$

**Routing prior and gradient flow.** The term  $\lambda \log w_{r,c'}$  acts as a learned prior over retained tokens and provides a differentiable path to the channel router despite the non-differentiable top- $k$  operations.  $\lambda$  is a learnable temperature parameter. Thus the pooling mechanism is trained not only through the compression branch but also through the downstream use of selected tokens.

### 3.5 Gating Mechanism

Learned gates combine the three branch outputs, following Yuan et al. (2025). Let  $\mathcal{B} = \{\text{u}, \text{c}, \text{s}\}$  index the branches. For each branch, an independent sigmoid gate  $g_{p,c}^b$  is computed from the embedded token. The gated branch outputs are then summed:

$$\mathbf{o}_{p,c} = \sum_{b \in \mathcal{B}} g_{p,c}^b \mathbf{o}_{p,c}^b. \quad (6)$$

### 3.6 Complexity Analysis

The univariate and compression branches attend over at most  $P$  temporal representations per query. The selection branch attends over at most  $k_p k_c$  retrieved tokens. With fixed  $k_p$  and  $k_c$ , NSTA scales linearly with the number of channels  $C$ .

Table 2: Attention-score complexity of representative interaction strategies. Projection and feed-forward costs are omitted; NSTA assumes retrieval budgets fixed independently of  $C$ .

| Method                       | Cross-channel access | Complexity             | Scaling in $C$ |
|------------------------------|----------------------|------------------------|----------------|
| PatchTST                     | None                 | $O(HCP^2d)$            | Linear         |
| Timer-XL with Time Attention | Dense                | $O(HC^2P^2d)$          | Quadratic      |
| Timer-XL with NSTA           | Sparse               | $O(HCP(P + k_p k_c)d)$ | Linear         |

## 4 Experiments

Native Sparse Time Attention is evaluated along four dimensions: forecasting performance relative to Time Attention (Section 4.1); computational scaling with respect to the number of variates (Section 4.2); the accuracy–compute trade-off induced by the retrieval budgets  $(k_p, k_c)$  (Section 4.3); and performance under long lookback windows, particularly on high-dimensional datasets. Full experimental details, including datasets, hyperparameters, retrieval budgets, metrics, and efficiency measurement protocols, are provided in Appendix C. Ablation studies are provided in Appendix D.3.

### 4.1 Forecasting Accuracy under Matched Configurations

NSTA is evaluated as a drop-in replacement for the Time Attention operator in Timer-XL (Liu et al., 2025), with the backbone, tokenisation, optimiser, and training schedule held fixed across configurations. The lookback window is set to 672, and we consider prediction horizons of 96, 192, 336, and 720. To avoid dataset-specific tuning, NSTA uses the same sparse configuration across all experiments, with  $k_p = 4$  and  $k_c = 32$ . Both Time Attention and NSTA are evaluated within the same Timer-XL decoder-only architecture, and results are averaged over three random seeds. Forecasting accuracy remains close to Time Attention on most benchmarks.

Table 3: Multivariate long-term forecasting under the Timer-XL one-for-all setting (Liu et al., 2025). MSE and MAE are averaged over prediction horizons with input length 672. NSTA uses  $k_p = 4$  and  $k_c = 32$  for all datasets. Full horizon-specific results are reported in Table 8.

| Models       | Timer-XL<br>(with NSTA) |              | Timer-XL<br>(2025) |              | Timer*<br>(2024c) |              | UniTST*<br>(2024a) |       | iTransformer*<br>(2024b) |       | DLinear*<br>(2023) |       | PatchTST*<br>(2023) |              | TimesNet*<br>(2022) |       | Stationary*<br>(2022) |       | Autoformer*<br>(2021) |       |
|--------------|-------------------------|--------------|--------------------|--------------|-------------------|--------------|--------------------|-------|--------------------------|-------|--------------------|-------|---------------------|--------------|---------------------|-------|-----------------------|-------|-----------------------|-------|
| Metric       | MSE                     | MAE          | MSE                | MAE          | MSE               | MAE          | MSE                | MAE   | MSE                      | MAE   | MSE                | MAE   | MSE                 | MAE          | MSE                 | MAE   | MSE                   | MAE   | MSE                   | MAE   |
| ECL          | 0.163                   | 0.254        | <b>0.156</b>       | <b>0.249</b> | 0.161             | 0.251        | 0.163              | 0.257 | 0.164                    | 0.258 | 0.165              | 0.265 | 0.169               | 0.268        | 0.201               | 0.303 | 0.265                 | 0.358 | 0.289                 | 0.379 |
| ETTh1        | <b>0.407</b>            | <b>0.431</b> | 0.415              | <b>0.434</b> | 0.418             | 0.436        | 0.429              | 0.447 | 0.421                    | 0.445 | 0.426              | 0.444 | 0.412               | 0.435        | 0.495               | 0.491 | 0.505                 | 0.513 | 0.517                 | 0.528 |
| Traffic      | 0.386                   | 0.261        | <b>0.380</b>       | <b>0.255</b> | 0.384             | 0.259        | 0.385              | 0.265 | <b>0.384</b>             | 0.274 | 0.423              | 0.298 | 0.391               | 0.275        | 0.602               | 0.322 | 0.630                 | 0.347 | 0.684                 | 0.433 |
| Weather      | 0.234                   | 0.272        | 0.242              | 0.276        | 0.232             | <b>0.270</b> | <b>0.231</b>       | 0.272 | 0.266                    | 0.291 | 0.239              | 0.291 | <b>0.226</b>        | <b>0.268</b> | 0.264               | 0.293 | 0.308                 | 0.329 | 0.435                 | 0.455 |
| Solar-Energy | <b>0.220</b>            | <b>0.259</b> | 0.230              | <b>0.256</b> | 0.233             | <b>0.249</b> | 0.241              | 0.275 | 0.213                    | 0.291 | 0.222              | 0.283 | <b>0.202</b>        | 0.269        | 0.213               | 0.295 | 0.254                 | 0.315 | 0.265                 | 0.325 |

\* Results taken from the Timer-XL paper (Liu et al., 2025).

## 4.2 Empirical Scaling with the Number of Channels

We evaluate empirical scaling by varying only the number of input channels  $C$ . NSTA uses fixed budgets  $k_p = 3$  and  $k_c = 128$ . We evaluate  $C \in \{256, 512, 768, 1024, 1536, 2048, 3072, 4096\}$  and report median inference and forward+backward latency. Implementation details are available in C.6.

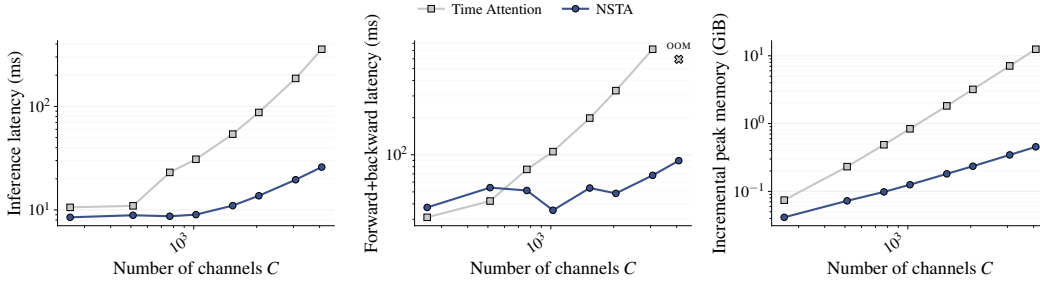


Figure 4: Empirical scaling with the number of input channels  $C$ . From left to right: inference latency, forward+backward latency, and incremental peak allocated memory for Time Attention and NSTA. OOM denotes configurations exceeding available device memory.

Figure 4 shows that NSTA becomes increasingly advantageous as dimensionality grows. While it incurs some overhead at small  $C$ , its latency scales substantially more slowly than Time Attention, which eventually exceeds available device memory. At  $C = 2048$ , NSTA is  $6.38\times$  faster at inference and  $6.81\times$  faster for forward+backward computation.

## 4.3 Accuracy–Efficiency Trade-off

We sweep the retrieval budgets ( $k_p, k_c$ ) and evaluate forecasting MSE at horizon 96 over three runs, together with measured inference latency. Implementation details are provided in Appendix C.6.

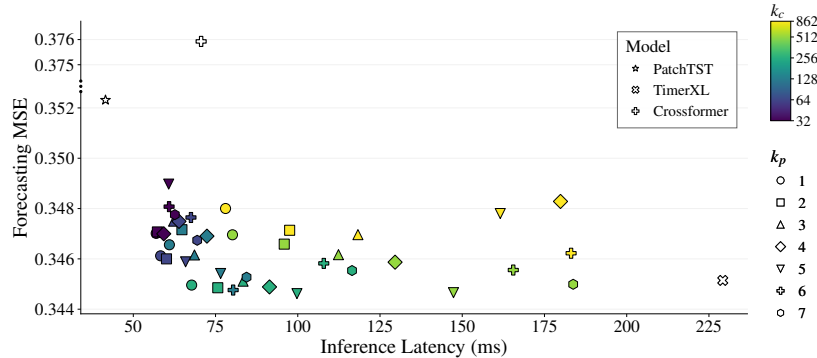


Figure 5: Accuracy–efficiency trade-off on the Traffic dataset ( $C = 862, P = 7$ ). Coloured markers denote configurations of NSTA (ours), with colour encoding the channel budget  $k_c$  and marker shape encoding the patch budget  $k_p$ . White markers denote the benchmarks models.

Dense Timer-XL provides the corresponding dense-attention operating point, while PatchTST serves as a channel-independent reference. We additionally compare against Crossformer (Zhang & Yan, 2023). All evaluated NSTA configurations outperform PatchTST on both datasets. Timer-XL with NSTA matches the forecasting accuracy of Timer-XL with Time Attention at horizon 96 while achieving  $3.4\times$  lower inference latency. Additional results characterising the accuracy–compute trade-off in measured GFLOPs on the Traffic and Global Temperature datasets are reported in Appendix D.

#### 4.4 Trading Sparsity for Context Length

The efficiency gains from NSTA allow longer input contexts on GTWSF-Temp ( $C = 3850$ ). Time Attention fits with a lookback of 48 but exceeds device memory at lookback 168 in our implementation (batch size 1, NVIDIA H100 80 GB, FlashAttention enabled), while NSTA remains feasible.

Table 4: GTWSF-Temp ( $C = 3850$ ). NSTA uses  $k_p = 3$ ,  $k_c = 256$ . Time Attention exceeds device memory at lookback 168. Mean  $\pm$  std over three seeds.

|                              | Lookback 48      |                  | Lookback 168                       |                                    |
|------------------------------|------------------|------------------|------------------------------------|------------------------------------|
|                              | MSE              | MAE              | MSE                                | MAE                                |
| Timer-XL with Time Attention | $7.17 \pm 0.011$ | $1.79 \pm 0.004$ | OOM                                |                                    |
| Timer-XL with NSTA           | $7.48 \pm 0.009$ | $1.83 \pm 0.001$ | <b><math>6.93 \pm 0.004</math></b> | <b><math>1.76 \pm 0.005</math></b> |
| PatchTST                     | $7.89 \pm 0.001$ | $1.89 \pm 0.002$ | $7.32 \pm 0.001$                   | $1.82 \pm 0.002$                   |

At lookback 48, NSTA is less accurate than Time Attention (7.48 vs. 7.17 MSE). At lookback 168, however, NSTA improves to 6.93 MSE, while dense attention is no longer feasible. Sparsity therefore trades some accuracy at fixed context for the ability to use a longer context, resulting in a better operating point under the same memory constraint.

#### 4.5 Analysis

We report the share of the layer’s output vector attributable to each branch. Figure 6 reveals a clear layer-dependent reallocation of contribution across the three branches on Traffic.

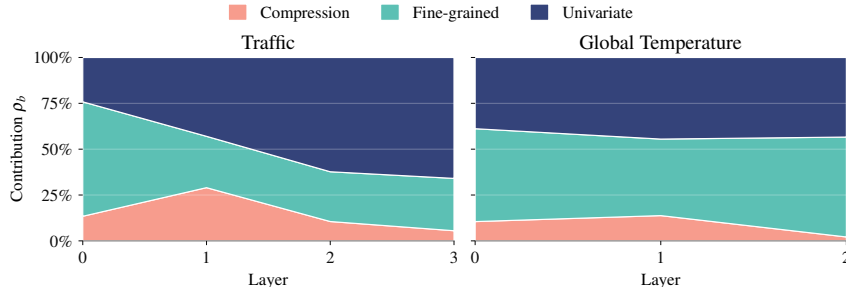


Figure 6: Layer-wise branch contributions on Traffic and Global Temperature.

The branch profiles vary across depth, indicating stage-specific rather than fixed contributions to the representation hierarchy. Differences between Traffic and Global Temperature further show that this allocation is data-dependent. Together, these results suggest that the three branches serve complementary roles whose relative importance varies with both network depth and dataset structure.

## 5 Conclusion and Future Work

The experiments suggest that sparsity is most useful in multivariate forecasting when viewed as a resource allocation mechanism rather than as a direct substitute for dense attention. At a fixed context length, reducing the number of cross-channel interactions can incur an accuracy cost. However, the resulting reduction in memory and computation makes larger channel sets and longer lookback windows feasible, which can lead to a better overall operating point.

The current evaluation is limited to supervised forecasting with fixed variable sets and architectures. Extending NSTA to transfer-learning and foundation-model settings, particularly across datasets with heterogeneous variable sets, is a natural direction for future work.

## 6 Ethics Statement

This work focuses on time-series forecasting. We do not identify any direct ethical risks specific to the proposed method.

## 7 Reproducibility Statement

The appendix provides the implementation details, including dataset descriptions, metrics, model configurations, and experimental protocols.

## 8 Acknowledgments

Acknowledgments omitted for double-blind review.

## References

- George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 5 edition, 2015. ISBN 9781118675021.
- Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2021. URL <https://arxiv.org/abs/2009.14794>.
- Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, 1993. ISSN 0304-405X. doi: [https://doi.org/10.1016/0304-405X\(93\)90023-5](https://doi.org/10.1016/0304-405X(93)90023-5). URL <https://www.sciencedirect.com/science/article/pii/0304405X93900235>.
- Kewei Hou. Industry information diffusion and the lead-lag effect in stock returns. *The Review of Financial Studies*, 20(4):1113–1138, 2007.
- Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL <https://arxiv.org/abs/1412.6980>.
- Jaebin Lee and Hankook Lee. Timeperceiver: An encoder-decoder framework for generalized time-series forecasting. In *Advances in Neural Information Processing Systems*, volume 38, 2025.
- Juncheng Liu, Chenghao Liu, Gerald Woo, Yiwei Wang, Bryan Hooi, Caiming Xiong, and Doyen Sahoo. Unitst: Effectively modeling inter-series and intra-series dependencies for multivariate time series forecasting, 2024a. URL <https://arxiv.org/abs/2406.04975>.
- Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35: 9881–9893, 2022.
- Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations (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 *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024c.
- Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer-XL: Long-context transformers for unified time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.04803.
- Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

- Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1185–1196, 2025. doi: 10.1145/3690624.3709325.
- James H. Stock and Mark W. Watson. Forecasting using principal components from a large number of predictors. *Journal of the American Statistical Association*, 97(460):1167–1179, 2002. doi: 10.1198/016214502388618960.
- Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity, 2020. URL <https://arxiv.org/abs/2006.04768>.
- 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*, volume 34, pp. 22419–22430. Curran Associates, Inc., 2021. URL [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/bcc0d400288793e8bdcd7c19a8ac0c2b-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/bcc0d400288793e8bdcd7c19a8ac0c2b-Paper.pdf).
- Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.
- Haixu Wu, Hang Zhou, Mingsheng Long, and Jianmin Wang. Interpretable weather forecasting for worldwide stations with a unified deep model. *Nature Machine Intelligence*, 5(6):602–611, 2023. URL <https://www.nature.com/articles/s42256-023-00667-9>.
- Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with 10k parameters, 2024. URL <https://arxiv.org/abs/2307.03756>.
- Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Y. X. Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025. arXiv:2502.11089.
- 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*, volume 37, pp. 11121–11128, 2023.
- Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2023.
- Lifan Zhao and Yanyan Shen. Rethinking channel dependence for multivariate time series forecasting: Learning from leading indicators, 2024. URL <https://arxiv.org/abs/2401.17548>.
- 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, 2021. URL <https://arxiv.org/abs/2012.07436>.

## A Related Work

**Temporal and cross-channel structure.** Multivariate forecasting architectures differ in the dependencies exposed to attention. PatchTST preserves fine-grained temporal structure within each channel but removes direct cross-channel interaction (Nie et al., 2023), while iTransformer attends across channel tokens after compressing temporal history into channel-level representations (Liu et al., 2024b). Crossformer, UniTST, and Timer-XL retain both temporal and channel structure more explicitly through two-stage, unified, or Time Attention over patch-channel representations (Zhang & Yan, 2023; Liu et al., 2024a,c, 2025). NSTA builds on this explicit patch-channel formulation but separates the resolution of temporal and cross-channel interactions: exact same-channel temporal dependencies are preserved, whereas cross-channel information is captured through compressed global context and selectively retained fine-grained interactions. Thus, NSTA retains the Timer-XL forecasting formulation while reducing interaction cost on the patch-channel grid.

**Efficient cross-channel interaction.** Several multivariate architectures reduce direct cross-channel interaction through intermediate routing structures. Crossformer communicates through learned router vectors (Zhang & Yan, 2023), UniTST uses a dispatcher mechanism (Liu et al., 2024a), and DUET learns sparse channel relations through frequency-domain soft clustering (Qiu et al., 2025). While these designs constrain cross-channel communication, DUET still constructs pairwise channel-relation matrices, retaining quadratic dependence on the number of channels. Unlike these approaches, NSTA combines compressed global communication with sparse direct access to the original patch-channel tokens. The compression branch captures broad cross-sectional structure at low cost, while the selection branch preserves fine-grained interactions for informative patches and channels.

**Sparse and efficient attention.** Efficient-attention methods reduce self-attention cost through low-rank projections (Wang et al., 2020), kernel feature maps (Choromanski et al., 2021), or sparse query selection (Zhou et al., 2021), but generally treat interactions over a one-dimensional token sequence uniformly. NSA instead decomposes attention into compression, selection, and local paths (Yuan et al., 2025). NSTA adapts this principle to multivariate forecasting, but its selection mechanism is reformulated for the two-dimensional patch-channel grid. Rather than selecting from a flat token sequence, NSTA performs structured two-stage retrieval over the temporal and channel dimensions, enabling sparse access to fine-grained  $P \times C$  interactions. Exact same-channel temporal attention replaces the local path, while compression summarizes cross-sectional context at each patch. By sparsifying only the costly cross-channel dimension, NSTA reduces attention complexity from quadratic to linear in the number of channels under fixed retrieval budgets.

**Channel identity and structural compatibility.** Architectures with an explicit channel dimension differ in how channel identity is encoded. PatchTST shares its encoder across channels, while iTransformer derives channel tokens from their observed histories (Nie et al., 2023; Liu et al., 2024b); other models, including Timer-XL and TimePerceiver, may incorporate variable-specific positional or channel information (Liu et al., 2025; Lee & Lee, 2025). NSTA instead encodes same- versus cross-channel relationships structurally: projections are shared across channels and routing is permutation symmetric along the channel axis. It therefore requires neither channel-indexed parameters nor a fixed channel ordering or vocabulary. Transfer across systems with different variable sets is left for future work.

## B Detailed Complexity Analysis

Let  $P$  be the number of temporal patches,  $C$  the number of channels,  $H$  the number of heads, and  $d$  the per-head dimension. We write  $\tilde{k}_p = \min(k_p, P)$  and  $\tilde{k}_c = \min(k_c, C)$  for the effective retrieval budgets, and suppress the batch dimension and the constant factors induced by causal masking. We count attention scores  $S$ ; the query–key cost is  $d \cdot S$  throughout, so the two scale identically.

**Score counts.** In Time Attention each of the  $PC$  queries attends to all  $PC$  keys, giving  $S_{\text{dense}} = O(HP^2C^2)$ . In NSTA, the univariate branch attends within each channel over the  $P$  patches and the compression branch provides one key–value pair per patch, each contributing  $O(HCP^2)$ ; the selection branch retrieves at most  $\tilde{k}_p$  patches and  $\tilde{k}_c$  channels within each, and this candidate set is shared by the  $C$  channel queries of the query patch, contributing  $O(HCP\tilde{k}_p\tilde{k}_c)$ . Hence

$$S_{\text{NSTA}} = O\left(HC(P^2 + P\tilde{k}_p\tilde{k}_c)\right). \quad (7)$$

For fixed  $P, H, d, k_p, k_c$  and  $C > k_c$  we have  $\tilde{k}_c = k_c$ , so both terms are linear in  $C$  against  $O(C^2)$  for dense attention. The separation depends on maintaining a bounded channel budget: if  $k_c \geq C$  then  $\tilde{k}_c = C$ , channel sparsification is disabled, and the selection term is quadratic again.

**Routing and projection overhead.** Routing adds no quadratic dependence on  $C$ . Pooling is linear in the number of input tokens. Patch routing is shared across the  $C$  queries of a patch and, under the default channel-aggregated formulation, costs  $O(HP^2d)$ ; channel routing is computed once per key patch and head and reused on retrieval, costing  $O(HPC)$ . Selection keys and values are projected only after channel selection, costing  $O(PH\tilde{k}_cDd)$  rather than projecting all  $PC$  tokens. All remaining token-wise projections are linear in  $C$ .

**Score-activation memory.** Score memory follows the counts above,  $O(S_{\text{NSTA}})$  against  $O(HP^2C^2)$ , and masking state obeys the same qualitative scaling. These are logical activation counts, not peak device memory, which additionally depends on the attention kernel, tensor materialization, checkpointing, and allocator behaviour; empirical peak memory is reported in Section 4.2.

## C Experimental Details

### C.1 Datasets

We evaluate on widely used multivariate forecasting benchmarks together with a high-dimensional benchmark used to probe the regime NSTA targets.

- **ETT** (Zhou et al., 2021) records seven variables associated with electricity transformers from July 2016 to July 2018. ETTh1 and ETTh2 are sampled hourly; ETTm1 and ETTm2 every 15 minutes.
- **ECL** (Wu et al., 2021) records the hourly electricity consumption of 321 clients.
- **Traffic** (Wu et al., 2021) contains hourly road occupancy rates from 862 sensors on San Francisco Bay Area highways between January 2015 and December 2016.
- **Weather** (Wu et al., 2021) contains 21 meteorological factors recorded every 10 minutes at the Max Planck Institute for Biogeochemistry weather station during 2020.
- **Solar-Energy** records the power production of 137 photovoltaic plants in 2006, sampled every 10 minutes.
- **GTWSF** (Wu et al., 2023) is drawn from the National Centers for Environmental Information and contains hourly average temperature and wind speed measurements from 3850 stations worldwide spanning 2019 to 2021. We use the temperature subset, denoted GTWSF-Temp, which provides  $C = 3850$  channels and is the highest-dimensional benchmark in our evaluation.

We follow the data processing and train/validation/test protocol of Wu et al. (2022), splitting chronologically to prevent leakage. Table 5 summarises the dimensions, split sizes, and forecasting settings.

Table 5: Dataset descriptions. *Dim.* is the number of channels  $C$ . *Length* gives the number of time points in the (train, validation, test) splits. *Setting* gives the input→output lengths.

| Dataset      | Domain (frequency)      | Dim. | Setting                                 | Length (train, val, test) |
|--------------|-------------------------|------|-----------------------------------------|---------------------------|
| ETTh1, ETTh2 | Electricity (hourly)    | 7    | $672 \rightarrow \{96, 192, 336, 720\}$ | (8545, 2881, 2881)        |
| ETTm1, ETTm2 | Electricity (15 min)    | 7    | $672 \rightarrow \{96, 192, 336, 720\}$ | (34465, 11521, 11521)     |
| ECL          | Electricity (hourly)    | 321  | $672 \rightarrow \{96, 192, 336, 720\}$ | (18317, 2633, 5261)       |
| Traffic      | Transportation (hourly) | 862  | $672 \rightarrow \{96, 192, 336, 720\}$ | (12185, 1757, 3509)       |
| Weather      | Climate (10 min)        | 21   | $672 \rightarrow \{96, 192, 336, 720\}$ | (36792, 5271, 10540)      |
| Solar-Energy | Energy (10 min)         | 137  | $672 \rightarrow \{96, 192, 336, 720\}$ | (36601, 5161, 10417)      |
| GTWSF-Temp   | Climate (hourly)        | 3850 | $48 \rightarrow 24$                     | (12280, 1755, 3509)       |

### C.2 Baselines

Our primary comparison is against Time Attention, obtained by running the Timer-XL architecture with its original attention operator under the same backbone, data pipeline, and training schedule; this isolates the effect of the attention operator. PatchTST (Nie et al., 2023) provides a channel-independent reference point without cross-channel interactions.

Results marked with \* in Tables 8 and 3 for Timer, UniTST, iTransformer, DLinear, PatchTST, TimesNet, Non-stationary Transformer, and Autoformer are reported by Liu et al. (2025) under the same one-for-all protocol and input length 672, and are reproduced here without modification. We did not rerun these baselines.

### C.3 Implementation Details

All experiments are implemented in PyTorch. We use the Adam optimiser (Kingma & Ba, 2017) with  $(\beta_1, \beta_2) = (0.9, 0.999)$  and the MSE training objective. Accuracy experiments are run on NVIDIA H100 GPUs; the latency and memory measurements are run on a single NVIDIA RTX A6000.

Because NSTA is evaluated as a drop-in replacement for Time Attention, we adopt the per-dataset model and optimisation configuration of Timer-XL without modification. These configurations are

listed in Table 6. No hyperparameter search was performed for NSTA beyond the sparsity budgets reported in Appendix C.4.

Table 6: Model and optimisation configurations.  $L$  is the number of Transformer blocks,  $D$  the residual width,  $d_k$  the per-head dimension,  $H$  the number of heads,  $P_{\text{size}}$  the patch size, and LR the initial learning rate. These follow Liu et al. (2025) and are shared by NSTA and dense Time Attention.

| Dataset      | $L$ | $D$  | $d_k$ | $H$ | $P_{\text{size}}$ | LR     | Batch size |
|--------------|-----|------|-------|-----|-------------------|--------|------------|
| ETTh1, ETTh2 | 1   | 1024 | 128   | 8   | 96                | 0.0001 | 32         |
| ETTh1, ETTm2 | 1   | 1024 | 128   | 8   | 96                | 0.0001 | 32         |
| ECL          | 5   | 512  | 64    | 8   | 96                | 0.0005 | 4          |
| Traffic      | 4   | 512  | 64    | 8   | 96                | 0.0005 | 4          |
| Weather      | 4   | 512  | 64    | 8   | 96                | 0.0005 | 32         |
| Solar-Energy | 6   | 512  | 64    | 8   | 96                | 0.0001 | 16         |
| GTWSF-Temp   | 3   | 1024 | 128   | 8   | 24                | 0.0001 | 8          |

Gates are initialised so that all three branches contribute equally at initialisation.

#### C.4 Sparsity Budgets

The retrieval budgets  $(k_p, k_c)$  are the only NSTA-specific hyperparameters. Table 7 records the budgets used in each experiment. In Section 4.1 we deliberately fix a single aggressive configuration across all datasets rather than tuning per dataset, so that the reported numbers measure the cost of uniform sparsification rather than the best achievable accuracy.

Table 7: Retrieval budgets by experiment.

| Experiment                                     | $k_p$             | $k_c$                                         |
|------------------------------------------------|-------------------|-----------------------------------------------|
| Matched configurations (Sec. 4.1, Tables 3, 8) | 4                 | 32                                            |
| Scaling benchmark (Sec. 4.2, Fig. 4)           | 3                 | 128                                           |
| Trade-off sweep, Traffic (Sec. 4.3, Fig. 7)    | $\{1, \dots, 7\}$ | $\{32, 64, 128, 256, 512, 862\}$              |
| Trade-off sweep, GTWSF-Temp (Sec. 4.3, Fig. 8) | $\{1, 2\}$        | $\{32, 64, 128, 256, 512, 1024, 2048, 3850\}$ |

Configurations with  $k_c = C$  disable channel sparsification and are included in the sweeps as the dense-channel limit discussed in Appendix B.

#### C.5 Metrics

We report mean squared error and mean absolute error, following prior work. For a univariate series  $X \in \mathbb{R}^T$  with prediction  $\hat{X}$ ,

$$\text{MSE} = \frac{1}{T} \sum_{i=1}^T (X_i - \hat{X}_i)^2, \quad \text{MAE} = \frac{1}{T} \sum_{i=1}^T |X_i - \hat{X}_i|. \quad (8)$$

For multivariate series both metrics are additionally averaged over the channel dimension. Table 3 reports the mean over the four horizons  $\{96, 192, 336, 720\}$ ; the accuracy–efficiency sweeps in Section 4.3 use the mean MSE over three seeds at horizon 96 only, so the two sets of numbers are not directly comparable.

#### C.6 Efficiency Measurement

**Latency and memory.** The scaling benchmark of Section 4.2 uses a controlled synthetic setting in which only the number of channels  $C$  varies. We use a three-layer model with  $D = 512$ ,  $d_{\text{ff}} = 2048$ , eight heads,  $P = 7$  patches, batch size one, and BF16 autocast, and sweep  $C \in \{256, 512, 768, 1024, 1536, 2048, 3072, 4096\}$ . We report the median over 300 timed executions following warm-up on a single NVIDIA RTX A6000.

For figure 5, we report wall-clock inference latency measured on a single NVIDIA RTX A6000 (PyTorch 2.6, CUDA 12.4). We measure the latency of one forward pass of the complete forecaster

in evaluation mode under `torch.no_grad`, on a batch of  $B = 4$  lookback windows spanning all  $C = 862$  Traffic channels, i.e. an input tensor of shape  $[4, 672, 862]$  mapped to a prediction of shape  $[4, 96, 862]$ . A single timed call therefore emits  $4 \times 96 \times 862 = 331,008$  predicted values. The measured region covers the entire model: patch embedding, all encoder layers, the output head and the instance de-normalisation. It excludes data loading and collation. We report the median of 50 timed iterations after 10 warm-up iterations, cycling over three distinct input batches, with a CUDA synchronisation at every timing boundary. All models are compiled with `torch.compile`.

**Attention GFLOPs.** The horizontal axis of Figures 7 and 8 counts floating-point operations in the attention operator of a single layer. For Time Attention this is the query-key and score-value cost over the full  $P \times C$  grid; for NSTA, it is the sum of the three branch costs of the mechanism, with routing and pooling. Isolating the operator this way makes the budgets comparable across configurations that share a backbone; absolute end-to-end cost is correspondingly higher for every point.

Attention GFLOPs are measured rather than derived analytically, using PyTorch’s `FlopCounterMode`, with the counter scoped to the attention operator of a single layer. The reported values therefore count each multiply-accumulate as two floating-point operations and cover the query-key, score-value, routing, and pooling matrix multiplications actually executed by each configuration.

### C.7 Ablation Protocol

The branch ablation of Section D.3 progressively enables the compression and selection branches on top of the univariate path, holding the backbone, retrieval budgets, and training schedule fixed. When a branch is disabled, its gate and projections are removed rather than zeroed, so the ablated models are strictly smaller. Because selection consumes the pooling weights produced by compression (Eq. 1), selection cannot be enabled without compression, and the univariate-plus-selection configuration is therefore not evaluated. Each configuration is trained with three random seeds; we report the mean and standard deviation.

### C.8 Gates Analysis

**Branch contribution.** The gate  $g_{p,v}^b$  alone does not measure how much the branch  $b$  contributes to the layer: it is a scalar applied to  $\mathbf{o}_{p,v}^b$ . We measure the contribution after the output projection  $\mathbf{W}^O$ , which distributes over Eq. 6:

$$\mathbf{y}_{p,v} = \mathbf{W}^O \mathbf{o}_{p,v} = \sum_{b \in \mathcal{B}} \underbrace{\mathbf{W}^O (g_{p,v}^b \mathbf{o}_{p,v}^b)}_{\mathbf{y}_{p,v}^b}, \quad (9)$$

We define the contribution of branch  $b$  as the projection of its term onto that vector:

$$\rho_{p,v}^b = \frac{\langle \mathbf{y}_{p,v}^b, \mathbf{y}_{p,v} \rangle}{\|\mathbf{y}_{p,v}\|^2}, \quad \sum_{b \in \mathcal{B}} \rho_{p,v}^b = 1, \quad (10)$$

We report  $\rho^b$  averaged over every token  $(p, v)$  and over three seeds.

## D Supplementary Results

### D.1 Multivariate Forecasting Results

Table 8: Full multivariate forecasting results. We perform rolling forecasts with a single model trained on each dataset, using a lookback length of 672 and prediction horizons in {96, 192, 336, 720}.

| Models            |     | Timer-XL<br>(with NSTA) |       | Timer-XL<br>(2025) |       | Timer*<br>(2024c) |       | UniTST*<br>(2024a) |       | iTransformer*<br>(2024b) |       | DLinear*<br>(2023) |       | PatchTST*<br>(2023) |       | TimesNet*<br>(2022) |       | Stationary*<br>(2022) |       | Autoformer*<br>(2021) |       |
|-------------------|-----|-------------------------|-------|--------------------|-------|-------------------|-------|--------------------|-------|--------------------------|-------|--------------------|-------|---------------------|-------|---------------------|-------|-----------------------|-------|-----------------------|-------|
| Metric            |     | MSE                     | MAE   | MSE                | MAE   | MSE               | MAE   | MSE                | MAE   | MSE                      | MAE   | MSE                | MAE   | MSE                 | MAE   | MSE                 | MAE   | MSE                   | MAE   | MSE                   | MAE   |
| ETTh1             | 96  | 0.371                   | 0.404 | 0.369              | 0.400 | 0.371             | 0.404 | 0.379              | 0.415 | 0.387                    | 0.418 | 0.369              | 0.400 | 0.373               | 0.403 | 0.452               | 0.463 | 0.452                 | 0.478 | 0.467                 | 0.499 |
|                   | 192 | 0.401                   | 0.424 | 0.411              | 0.427 | 0.407             | 0.429 | 0.415              | 0.438 | 0.416                    | 0.437 | 0.405              | 0.422 | 0.405               | 0.425 | 0.474               | 0.477 | 0.484                 | 0.510 | 0.492                 | 0.523 |
|                   | 336 | 0.415                   | 0.436 | 0.433              | 0.442 | 0.434             | 0.445 | 0.440              | 0.454 | 0.434                    | 0.450 | 0.435              | 0.445 | 0.423               | 0.440 | 0.493               | 0.489 | 0.511                 | 0.522 | 0.519                 | 0.531 |
|                   | 720 | 0.441                   | 0.461 | 0.449              | 0.466 | 0.441             | 0.466 | 0.482              | 0.482 | 0.447                    | 0.473 | 0.493              | 0.508 | 0.445               | 0.471 | 0.560               | 0.534 | 0.571                 | 0.543 | 0.589                 | 0.560 |
|                   | Avg | 0.407                   | 0.431 | 0.415              | 0.434 | 0.418             | 0.436 | 0.429              | 0.447 | 0.421                    | 0.445 | 0.426              | 0.444 | 0.412               | 0.435 | 0.495               | 0.491 | 0.505                 | 0.513 | 0.517                 | 0.528 |
| ETTh2             | 96  | 0.297                   | 0.351 | 0.298              | 0.354 | 0.285             | 0.344 | 0.343              | 0.398 | 0.304                    | 0.362 | 0.305              | 0.371 | 0.289               | 0.347 | 0.340               | 0.374 | 0.348                 | 0.403 | 0.358                 | 0.397 |
|                   | 192 | 0.356                   | 0.391 | 0.367              | 0.400 | 0.365             | 0.400 | 0.376              | 0.420 | 0.372                    | 0.407 | 0.412              | 0.439 | 0.360               | 0.393 | 0.402               | 0.414 | 0.408                 | 0.448 | 0.435                 | 0.451 |
|                   | 336 | 0.370                   | 0.410 | 0.399              | 0.429 | 0.412             | 0.440 | 0.399              | 0.435 | 0.418                    | 0.440 | 0.527              | 0.508 | 0.389               | 0.420 | 0.452               | 0.452 | 0.424                 | 0.457 | 0.454                 | 0.475 |
|                   | 720 | 0.432                   | 0.460 | 0.447              | 0.470 | 0.468             | 0.487 | 0.419              | 0.457 | 0.463                    | 0.476 | 0.830              | 0.653 | 0.398               | 0.440 | 0.462               | 0.468 | 0.448                 | 0.476 | 0.479                 | 0.492 |
|                   | Avg | 0.364                   | 0.403 | 0.378              | 0.413 | 0.382             | 0.418 | 0.384              | 0.428 | 0.389                    | 0.421 | 0.518              | 0.493 | 0.359               | 0.400 | 0.414               | 0.427 | 0.407                 | 0.446 | 0.431                 | 0.454 |
| ETTm1             | 96  | 0.302                   | 0.347 | 0.299              | 0.349 | 0.281             | 0.338 | 0.289              | 0.348 | 0.311                    | 0.365 | 0.307              | 0.350 | 0.285               | 0.346 | 0.338               | 0.375 | 0.414                 | 0.414 | 0.466                 | 0.466 |
|                   | 192 | 0.354                   | 0.376 | 0.354              | 0.380 | 0.330             | 0.368 | 0.332              | 0.375 | 0.353                    | 0.390 | 0.337              | 0.368 | 0.329               | 0.372 | 0.371               | 0.387 | 0.524                 | 0.482 | 0.504                 | 0.496 |
|                   | 336 | 0.390                   | 0.399 | 0.394              | 0.404 | 0.367             | 0.393 | 0.365              | 0.397 | 0.387                    | 0.411 | 0.366              | 0.387 | 0.363               | 0.394 | 0.410               | 0.411 | 0.541                 | 0.497 | 0.574                 | 0.530 |
|                   | 720 | 0.449                   | 0.435 | 0.463              | 0.443 | 0.432             | 0.433 | 0.421              | 0.431 | 0.452                    | 0.445 | 0.419              | 0.419 | 0.421               | 0.426 | 0.478               | 0.450 | 0.578                 | 0.509 | 0.596                 | 0.558 |
|                   | Avg | 0.374                   | 0.389 | 0.378              | 0.394 | 0.352             | 0.383 | 0.352              | 0.388 | 0.376                    | 0.403 | 0.357              | 0.381 | 0.349               | 0.385 | 0.399               | 0.406 | 0.514                 | 0.475 | 0.535                 | 0.512 |
| ETTm2             | 96  | 0.182                   | 0.264 | 0.182              | 0.265 | 0.175             | 0.257 | 0.171              | 0.260 | 0.183                    | 0.272 | 0.167              | 0.263 | 0.172               | 0.259 | 0.187               | 0.267 | 0.237                 | 0.306 | 0.255                 | 0.339 |
|                   | 192 | 0.249                   | 0.309 | 0.242              | 0.306 | 0.239             | 0.301 | 0.228              | 0.230 | 0.250                    | 0.315 | 0.230              | 0.311 | 0.233               | 0.299 | 0.249               | 0.309 | 0.330                 | 0.387 | 0.279                 | 0.335 |
|                   | 336 | 0.301                   | 0.343 | 0.296              | 0.342 | 0.293             | 0.342 | 0.282              | 0.336 | 0.311                    | 0.356 | 0.298              | 0.361 | 0.280               | 0.331 | 0.321               | 0.351 | 0.404                 | 0.424 | 0.331                 | 0.374 |
|                   | 720 | 0.389                   | 0.400 | 0.382              | 0.395 | 0.392             | 0.407 | 0.380              | 0.398 | 0.417                    | 0.419 | 0.432              | 0.446 | 0.357               | 0.382 | 0.497               | 0.403 | 0.525                 | 0.486 | 0.413                 | 0.450 |
|                   | Avg | 0.280                   | 0.329 | 0.275              | 0.327 | 0.275             | 0.327 | 0.265              | 0.306 | 0.290                    | 0.340 | 0.282              | 0.345 | 0.261               | 0.318 | 0.314               | 0.333 | 0.374                 | 0.401 | 0.320                 | 0.374 |
| ECL               | 96  | 0.128                   | 0.221 | 0.127              | 0.220 | 0.129             | 0.221 | 0.130              | 0.225 | 0.133                    | 0.229 | 0.138              | 0.238 | 0.132               | 0.232 | 0.184               | 0.288 | 0.185                 | 0.287 | 0.256                 | 0.357 |
|                   | 192 | 0.149                   | 0.241 | 0.144              | 0.237 | 0.148             | 0.239 | 0.150              | 0.244 | 0.158                    | 0.258 | 0.152              | 0.251 | 0.151               | 0.250 | 0.192               | 0.295 | 0.282                 | 0.368 | 0.291                 | 0.376 |
|                   | 336 | 0.167                   | 0.259 | 0.159              | 0.253 | 0.164             | 0.256 | 0.166              | 0.262 | 0.168                    | 0.262 | 0.167              | 0.268 | 0.171               | 0.272 | 0.200               | 0.303 | 0.289                 | 0.377 | 0.290                 | 0.379 |
|                   | 720 | 0.207                   | 0.295 | 0.193              | 0.284 | 0.201             | 0.289 | 0.206              | 0.297 | 0.205                    | 0.294 | 0.203              | 0.302 | 0.222               | 0.318 | 0.228               | 0.325 | 0.305                 | 0.399 | 0.320                 | 0.403 |
|                   | Avg | 0.163                   | 0.254 | 0.156              | 0.249 | 0.161             | 0.251 | 0.163              | 0.257 | 0.164                    | 0.258 | 0.165              | 0.265 | 0.169               | 0.268 | 0.201               | 0.303 | 0.265                 | 0.358 | 0.289                 | 0.379 |
| Traffic           | 96  | 0.347                   | 0.240 | 0.345              | 0.237 | 0.348             | 0.240 | 0.359              | 0.250 | 0.353                    | 0.259 | 0.399              | 0.285 | 0.359               | 0.255 | 0.593               | 0.315 | 0.610                 | 0.322 | 0.675                 | 0.412 |
|                   | 192 | 0.369                   | 0.251 | 0.366              | 0.247 | 0.369             | 0.250 | 0.373              | 0.257 | 0.373                    | 0.267 | 0.409              | 0.290 | 0.377               | 0.265 | 0.596               | 0.317 | 0.626                 | 0.346 | 0.679                 | 0.423 |
|                   | 336 | 0.387                   | 0.262 | 0.383              | 0.256 | 0.388             | 0.260 | 0.386              | 0.265 | 0.386                    | 0.275 | 0.422              | 0.297 | 0.393               | 0.276 | 0.600               | 0.319 | 0.633                 | 0.352 | 0.688                 | 0.440 |
|                   | 720 | 0.439                   | 0.292 | 0.426              | 0.281 | 0.431             | 0.285 | 0.421              | 0.286 | 0.425                    | 0.296 | 0.461              | 0.319 | 0.436               | 0.305 | 0.619               | 0.335 | 0.651                 | 0.366 | 0.693                 | 0.457 |
|                   | Avg | 0.386                   | 0.261 | 0.380              | 0.255 | 0.384             | 0.259 | 0.385              | 0.265 | 0.384                    | 0.274 | 0.423              | 0.298 | 0.391               | 0.275 | 0.602               | 0.322 | 0.630                 | 0.347 | 0.684                 | 0.433 |
| Weather           | 96  | 0.155                   | 0.206 | 0.160              | 0.210 | 0.151             | 0.202 | 0.152              | 0.206 | 0.174                    | 0.225 | 0.169              | 0.229 | 0.149               | 0.202 | 0.169               | 0.228 | 0.185                 | 0.241 | 0.355                 | 0.409 |
|                   | 192 | 0.201                   | 0.250 | 0.208              | 0.254 | 0.196             | 0.245 | 0.198              | 0.249 | 0.227                    | 0.268 | 0.211              | 0.268 | 0.194               | 0.245 | 0.222               | 0.269 | 0.286                 | 0.325 | 0.421                 | 0.450 |
|                   | 336 | 0.254                   | 0.290 | 0.262              | 0.295 | 0.249             | 0.288 | 0.251              | 0.291 | 0.290                    | 0.309 | 0.258              | 0.306 | 0.244               | 0.285 | 0.290               | 0.310 | 0.323                 | 0.347 | 0.452                 | 0.465 |
|                   | 720 | 0.328                   | 0.342 | 0.338              | 0.347 | 0.330             | 0.344 | 0.322              | 0.340 | 0.374                    | 0.360 | 0.320              | 0.362 | 0.317               | 0.338 | 0.376               | 0.364 | 0.436                 | 0.401 | 0.513                 | 0.496 |
|                   | Avg | 0.234                   | 0.272 | 0.242              | 0.276 | 0.232             | 0.270 | 0.231              | 0.272 | 0.266                    | 0.291 | 0.239              | 0.291 | 0.226               | 0.268 | 0.264               | 0.293 | 0.308                 | 0.329 | 0.435                 | 0.455 |
| Solar-Energy      | 96  | 0.186                   | 0.230 | 0.182              | 0.224 | 0.212             | 0.230 | 0.190              | 0.240 | 0.183                    | 0.265 | 0.193              | 0.258 | 0.168               | 0.237 | 0.180               | 0.272 | 0.199                 | 0.290 | 0.206                 | 0.296 |
|                   | 192 | 0.212                   | 0.251 | 0.214              | 0.246 | 0.232             | 0.246 | 0.223              | 0.264 | 0.205                    | 0.283 | 0.214              | 0.274 | 0.189               | 0.257 | 0.199               | 0.286 | 0.243                 | 0.307 | 0.254                 | 0.328 |
|                   | 336 | 0.226                   | 0.266 | 0.239              | 0.264 | 0.237             | 0.253 | 0.250              | 0.283 | 0.224                    | 0.299 | 0.233              | 0.291 | 0.212               | 0.277 | 0.220               | 0.301 | 0.264                 | 0.322 | 0.272                 | 0.330 |
|                   | 720 | 0.257                   | 0.290 | 0.287              | 0.292 | 0.292             | 0.266 | 0.292              | 0.311 | 0.239                    | 0.316 | 0.246              | 0.307 | 0.240               | 0.305 | 0.251               | 0.321 | 0.310                 | 0.339 | 0.326                 | 0.347 |
|                   | Avg | 0.220                   | 0.259 | 0.230              | 0.256 | 0.233             | 0.249 | 0.241              | 0.275 | 0.213                    | 0.291 | 0.222              | 0.283 | 0.202               | 0.269 | 0.213               | 0.295 | 0.254                 | 0.315 | 0.265                 | 0.325 |
| First-place count |     | 6                       | 5     | 10                 | 13    | 2                 | 10    | 2                  | 2     | 1                        | 0     | 3                  | 6     | 17                  | 9     | 0                   | 0     | 0                     | 0     | 0                     | 0     |

\* Results taken from the Timer-XL paper (Liu et al., 2025).

## D.2 Accuracy-Efficiency Trade-off

Accuracy improves with the channel budget, from 7.62 at  $k_c = 32$  ( $\approx 102$  GFLOPs) to 7.20 at  $k_c = C$  ( $\approx 377$  GFLOPs), while  $k_p \in 1, 2$  has little effect. Dense Timer-XL remains best (7.17,  $\approx 307$  GFLOPs), but NSTA progressively approaches dense accuracy as compute increases:  $k_c = 2048$  reaches 7.26 at  $\approx 248$  GFLOPs.

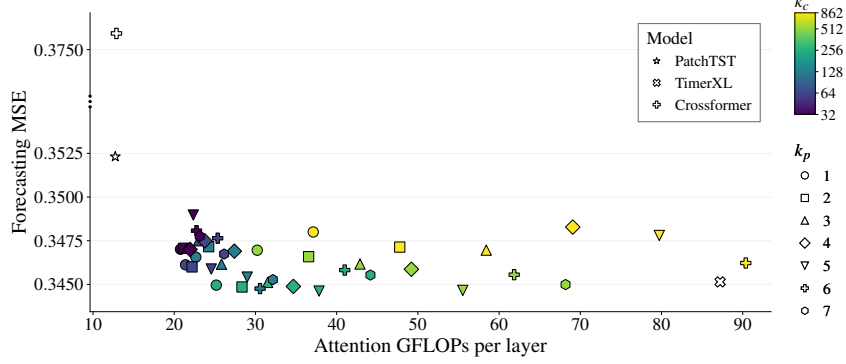


Figure 7: Accuracy–efficiency trade-off on the Traffic dataset ( $C = 862$ ,  $P = 7$ ). Coloured markers denote configurations of NSTA (ours), with colour encoding the channel budget  $k_c$  and marker shape encoding the patch budget  $k_p$ . White markers denote the benchmarks models.

On Traffic ( $C = 862$ , Figure 7), the best NSTA configurations reach  $\text{MSE} \approx 0.3447$  at 30–37 attention GFLOPs, matching Timer-XL (0.3452 at  $\approx 87$  GFLOPs) at roughly one third of the attention cost.

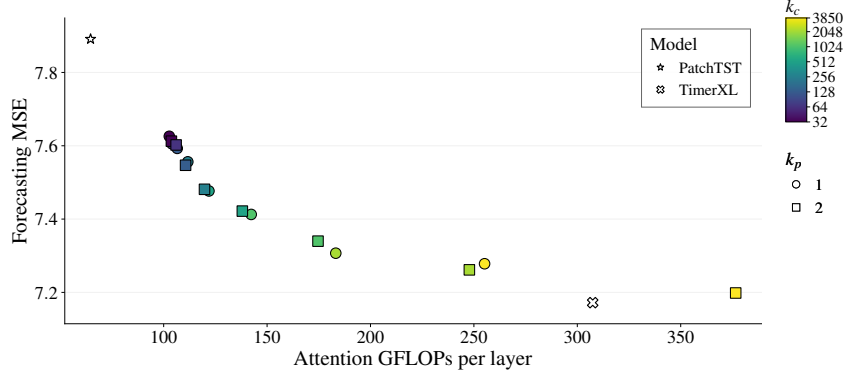


Figure 8: Accuracy–efficiency trade-off on the GTWSF-Temp dataset ( $C = 3850$ ,  $P = 2$ ). Accuracy improves monotonically with the channel budget  $k_c$ , tracing a controllable operating curve rather than an interior optimum.

NSTA offers lower-compute operating points with a controlled accuracy trade-off, yielding a dataset-dependent accuracy–compute frontier rather than uniformly dominating dense Time Attention.

## D.3 Ablation Studies

The compression and selection branches are progressively added to the univariate path (Table 9). The full model achieves the lowest MSE on Weather, Traffic, and GTWSF-Temp, while matching the univariate baseline on Solar within run-to-run variation. Compression alone does not consistently improve performance, whereas adding selection recovers or improves accuracy across all datasets, consistent with complementary roles for coarse cross-channel context and fine-grained retrieval.

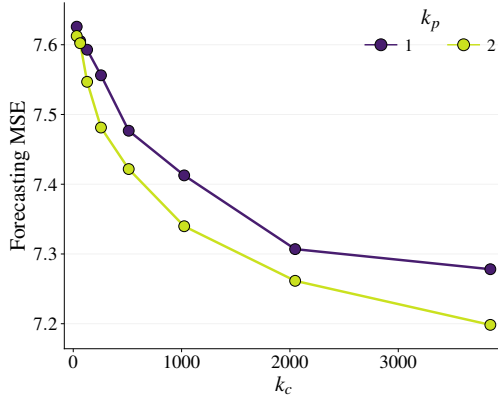


Figure 9: Forecasting MSE as a function of the channel retrieval budget  $k_c$  for  $k_p \in \{1, 2\}$ . Increasing  $k_c$  consistently improves accuracy, while the effect of  $k_p$  is comparatively small.

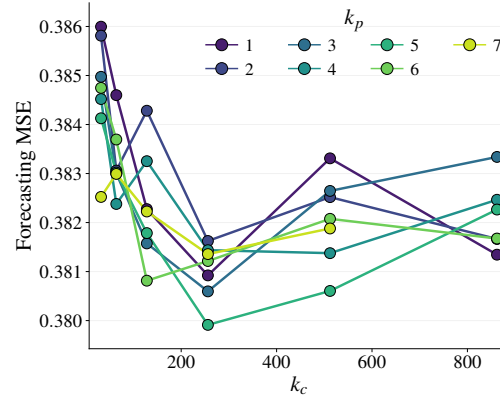


Figure 10: Forecasting MSE as a function of the channel retrieval budget  $k_c$  across  $k_p \in \{1, \dots, 7\}$ . Performance depends only weakly on  $k_p$  and exhibits a non-monotonic trade-off with  $k_c$ .

Table 9: Ablation of the NSTA branches. Results are mean  $\pm$  standard deviation over three runs; lower is better. Lookback window is 672 for Solar, Weather and Traffic and 48 for GTWSF-Temp. Selection branches use  $k_p = 4$ ,  $k_c = 32$  on Traffic, Solar and Weather datasets and  $k_p = 3$  and  $k_c = 256$  for GTWSF-Temp.

| Dataset    | Branches |       |         | Metrics                             |                                     |
|------------|----------|-------|---------|-------------------------------------|-------------------------------------|
|            | Uni.     | Comp. | Select. | MSE $\downarrow$                    | MAE $\downarrow$                    |
| Solar      | ✓        |       |         | <b>0.220 <math>\pm</math> 0.015</b> | <b>0.259 <math>\pm</math> 0.012</b> |
|            | ✓        | ✓     |         | 0.232 $\pm$ 0.010                   | 0.273 $\pm$ 0.008                   |
|            | ✓        | ✓     | ✓       | 0.219 $\pm$ 0.016                   | 0.261 $\pm$ 0.007                   |
| Weather    | ✓        |       |         | 0.245 $\pm$ 0.007                   | 0.279 $\pm$ 0.005                   |
|            | ✓        | ✓     |         | 0.245 $\pm$ 0.004                   | 0.277 $\pm$ 0.001                   |
|            | ✓        | ✓     | ✓       | <b>0.234 <math>\pm</math> 0.003</b> | <b>0.272 <math>\pm</math> 0.001</b> |
| Traffic    | ✓        |       |         | 0.389 $\pm$ 0.002                   | 0.265 $\pm$ 0.003                   |
|            | ✓        | ✓     |         | 0.396 $\pm$ 0.008                   | 0.269 $\pm$ 0.004                   |
|            | ✓        | ✓     | ✓       | <b>0.382 <math>\pm</math> 0.002</b> | <b>0.258 <math>\pm</math> 0.002</b> |
| GTWSF-Temp | ✓        |       |         | 7.828 $\pm$ 0.003                   | 1.874 $\pm$ 0.001                   |
|            | ✓        | ✓     |         | 7.612 $\pm$ 0.015                   | 1.849 $\pm$ 0.002                   |
|            | ✓        | ✓     | ✓       | <b>7.481 <math>\pm</math> 0.009</b> | <b>1.831 <math>\pm</math> 0.001</b> |

To test whether the ranking itself carries signal rather than merely acting as a regulariser, ranked selection is compared with uniform random selection of the same number of fine-grained interactions. The random subset is resampled at every forward pass, with all other components, hyperparameters, and compute budgets held fixed (Table 10). Ranked selection performs better on all three datasets.

Table 10: Ablation study on the selection strategy. Results are reported as mean  $\pm$  standard deviation across runs. Lower is better. Lookback is 672 for Traffic and ECL, 168 for GTWSF-Temp.

| Dataset    | $k_p$ | $k_c$ | Selection Strategy | MSE $\downarrow$                    | MAE $\downarrow$                    |
|------------|-------|-------|--------------------|-------------------------------------|-------------------------------------|
| Traffic    | 2     | 32    | Top-ranked         | <b>0.386 <math>\pm</math> 0.003</b> | <b>0.262 <math>\pm</math> 0.002</b> |
|            |       |       | Random             | 0.413 $\pm$ 0.010                   | 0.274 $\pm$ 0.008                   |
| ECL        | 2     | 32    | Top-ranked         | <b>0.160 <math>\pm</math> 0.001</b> | <b>0.251 <math>\pm</math> 0.002</b> |
|            |       |       | Random             | 0.163 $\pm$ 0.001                   | 0.254 $\pm$ 0.003                   |
| GTWSF-Temp | 2     | 256   | Top-ranked         | <b>6.989 <math>\pm</math> 0.048</b> | <b>1.767 <math>\pm</math> 0.004</b> |
|            |       |       | Random             | 7.079 $\pm$ 0.124                   | 1.778 $\pm$ 0.016                   |

The selection-branch routing prior is further ablated on Traffic across five values of  $k_c$ , with  $k_p = 4$ . Enabling the prior improves MSE at every budget, with a mean reduction of 2.1% across matched runs (Table 11).

Table 11: Effect of the selection-branch routing prior on Traffic Dataset. Each entry is the mean over the  $\{96, 192, 336, 720\}$  rolling horizons, averaged over three seeds.  $\Delta$  is the paired MSE improvement from enabling the bias. All configurations use  $k_p = 4$  and lookback 672.

| $k_c$ | No prior            |                     | Prior                                 |                                       | $\Delta$ |     |
|-------|---------------------|---------------------|---------------------------------------|---------------------------------------|----------|-----|
|       | MSE                 | MAE                 | MSE                                   | MAE                                   | MSE      | %   |
| 32    | $0.3922 \pm 0.0039$ | $0.2637 \pm 0.0030$ | <b><math>0.3857 \pm 0.0024</math></b> | <b><math>0.2576 \pm 0.0013</math></b> | 0.0095   | 2.5 |
| 64    | $0.3903 \pm 0.0034$ | $0.2618 \pm 0.0012$ | <b><math>0.3852 \pm 0.0020</math></b> | <b><math>0.2607 \pm 0.0038</math></b> | 0.0051   | 1.3 |
| 128   | $0.3897 \pm 0.0041$ | $0.2611 \pm 0.0016$ | <b><math>0.3817 \pm 0.0013</math></b> | <b><math>0.2564 \pm 0.0003</math></b> | 0.0080   | 2.1 |
| 256   | $0.3923 \pm 0.0045$ | $0.2618 \pm 0.0029$ | <b><math>0.3808 \pm 0.0023</math></b> | <b><math>0.2565 \pm 0.0010</math></b> | 0.0115   | 3.0 |
| 512   | $0.3891 \pm 0.0025$ | $0.2620 \pm 0.0031$ | <b><math>0.3822 \pm 0.0009</math></b> | <b><math>0.2561 \pm 0.0006</math></b> | 0.0069   | 1.8 |
| Avg.  | 0.3907              | 0.2621              | <b>0.3825</b>                         | <b>0.2575</b>                         | 0.0082   | 2.1 |

Enabling the score bias improves Traffic MSE in all 15 matched runs (five channel budgets  $\times$  three seeds), by 0.0082 on average (2.1%), and MAE in 14/15 runs by 0.0046 on average.

#### D.4 Analysis

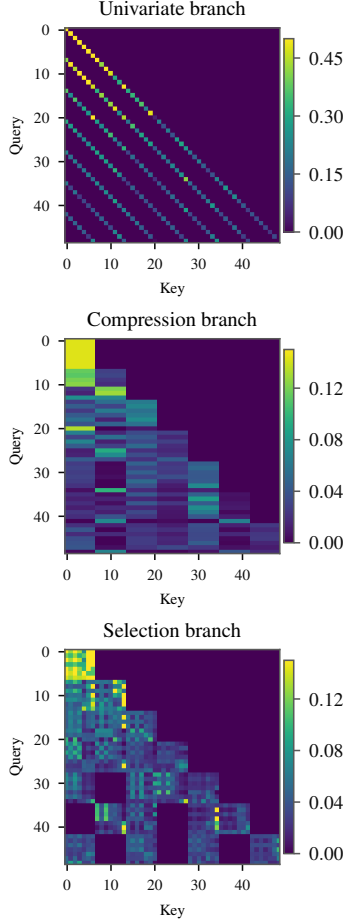


Figure 11: Attention maps for the three branches on ETTh1 dataset.