GRAIL: GRANGER-INSPIRED CAUSAL MASKING ENHANCED MULTIVARIATE TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate time series forecasting is essential for a wide range of real-world applications, including finance, energy, climate analysis, and transportation. However, accurate and interpretable forecasting remains challenging due to heterogeneous temporal dynamics and complex cross-variable dependencies. Existing approaches often rely on fixed temporal modeling strategies or dense channel interactions, making it difficult to capture sparse, directional, and delayed predictive relationships among variables. In this work, we propose GRAIL, a framework that jointly models adaptive temporal patterns and lag-aware cross-channel dependencies for multivariate forecasting. The proposed method introduces a Temporal Scale Fusion module to integrate short-, base-, and long-range temporal representations through sample-level adaptive weighting. It further incorporates a Granger-inspired dependency modeling module that estimates multi-step lagged predictive relationships and constructs sparse masks to guide cross-variable information aggregation. By combining adaptive temporal representation learning with structure-aware dependency modeling, the proposed framework improves forecasting accuracy while providing enhanced interpretability. Extensive experiments on eight real-world benchmark datasets demonstrate that GRAIL achieves state-of-the-art performance.

1 INTRODUCTION

Multivariate time series forecasting (MTSF) plays an important role in modeling the evolution of complex systems, with applications spanning financial markets (Huang et al., 2022; Sezer et al., 2020), energy management (Sun et al., 2022; Wang et al., 2024a), climate analysis, and traffic optimization (Yang et al., 2022; Kieu et al., 2024). By leveraging historical observations from multiple variables, MTSF aims to predict future system states and support downstream decision-making, including risk management, resource scheduling, and anomaly detection. For each target variable, useful predictive information may come from both its own history and the histories of other variables, and the relevance of these signals can change across input windows. A forecasting model must therefore decide which temporal patterns to retain and which cross-channel information to use. Despite recent advances in deep forecasting models, making these decisions accurately and transparently remains challenging because of heterogeneous temporal dynamics and complex cross-channel dependencies.

First, real-world time series are often non-stationary and exhibit diverse temporal patterns. Different samples or variables may be dominated by short-term fluctuations, periodic variations, or long-term trends under changing environments and system states (Kim et al., 2021; Liu et al., 2025). These patterns can coexist within the same input window, while their relative predictive importance may change from one sample to another. Existing MTSF methods have explored various temporal modeling strategies, including decomposition-based architectures (Wu et al., 2021; Wang et al., 2024b), patch-based models (Nie et al., 2022; Chen et al., 2023), frequency-domain approaches (Wu et al., 2022; Cai et al., 2024), and Transformer-based forecasters (Zhang & Yan, 2022; Liu et al., 2024). Although these approaches capture useful temporal structures, a fixed choice of scales or fusion weights may emphasize patterns that are less informative for a particular input. This motivates adapting the contribution of different temporal scales to each sample.

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.

Second, cross-channel dependencies in multivariate systems are often sparse, directional, and delayed. As illustrated in Figure 1, existing channel interaction strategies vary significantly in how they capture such dependencies. Channel-independent models, such as DLinear Zeng et al. (2023) and PatchTST Nie et al. (2022), reduce noisy interactions but may overlook informative cross-variable relationships. Channel-dependent models, including Crossformer Zhang & Yan (2022), iTransformer Liu et al. (2024), SOFTS Han et al. (2024), and DUET Qiu et al. (2025), capture variable interactions through attention, mixing, or clustering mechanisms, but dense aggregation can introduce redundant or spurious dependencies. Recent causal-aware forecasting methods, such as Causalformer Zhang et al. (2023), CAIFormer Zhang et al. (2026), and AR-Causal Zhou et al. (2026), incorporate causal graphs or masks to guide dependency modeling. However, these approaches often depend on explicit causal structure estimation, predefined assumptions, or additional inference procedures, limiting their flexibility in modeling input-dependent and lag-varying interactions. Therefore, developing a lightweight, input-dependent, and lag-aware predictive mask that can be seamlessly integrated with adaptive temporal modeling remains an open problem.

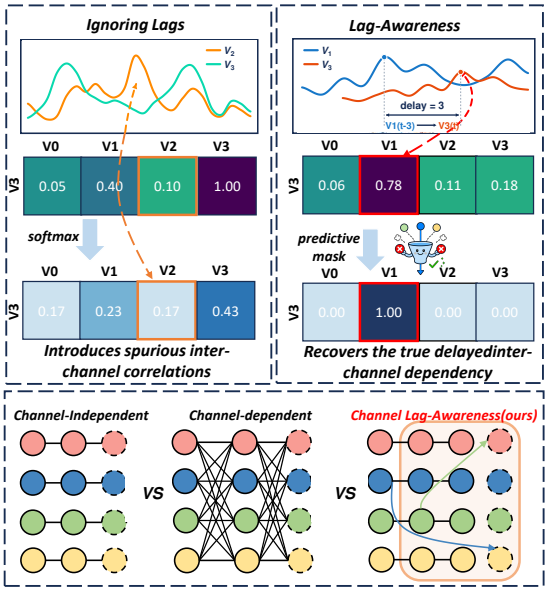


Figure 1: Comparison of channel interaction strategies. The upper panels show that ignoring lags can spread attention across less informative channels, while GRAIL's lag-aware mask selects a delayed predictive source. The lower panel contrasts channel-independent, dense, and sparse lag-aware information flow.

To address these challenges, we propose GRAIL, a Granger-inspired framework for multivariate time series forecasting. Specifically, the Temporal Scale Fusion (TSF) module captures short-, base-, and long-scale temporal representations and adaptively integrates them through sample-level gating, while the Granger-Inspired Causal Mask (GICM) module estimates multi-step lagged predictive dependencies from raw observations and generates sparse forecasting-oriented masks to guide cross-channel attention. By combining temporal-scale adaptation with sparse lag-aware channel aggregation, GRAIL reduces the influence of noisy interactions while emphasizing informative source variables. Extensive experiments on eight real-world benchmark datasets demonstrate that GRAIL achieves competitive or superior performance, and further analyses validate the effectiveness of its key components.

Our contributions are summarized as follows:

- We propose GRAIL, a forecasting framework that jointly models sample-adaptive temporal patterns and sparse lagged cross-channel predictive dependencies.
- We design the TSF module that adaptively integrates short-, base-, and long-scale temporal representations through sample-level gating.
- We introduce the GICM module that constructs sparse forecasting-oriented predictive masks based on multi-step lagged dependencies.
- Extensive experiments demonstrate that GRAIL achieves state-of-the-art performance on multiple benchmark datasets.

2 Related Work

Multivariate Time Series Forecasting Models MTSF predicts future values from the historical observations of multiple variables. Existing methods use different temporal representations: Autoformer (Wu et al., 2021) and FEDformer (Zhou et al., 2022) model decomposed trend and seasonal

components; HASPFormer (Ljubić et al., 2026) uses an attention-based architecture; and TimesNet (Wu et al., 2022) and MSGNet (Cai et al., 2024) capture periodic or multi-scale patterns. More recent methods address scale variation and non-stationarity through scale mixing, distribution adaptation, or temporal clustering (Wang et al., 2024b; Liu et al., 2025; Qiu et al., 2025). These approaches provide complementary ways to represent temporal dynamics. Determining how much each scale should contribute for a particular input sample remains an important design choice.

Channel Dependency Modeling in MTSF Channel-independent methods, including DLinear (Zeng et al., 2023) and PatchTST (Nie et al., 2022), model each variable separately, avoiding interference from unrelated channels but missing potentially useful interactions. Channel-dependent methods, such as MTGNN (Wu et al., 2020), Crossformer (Zhang & Yan, 2022), iTransformer (Liu et al., 2024), and SOFTS (Han et al., 2024), model cross-variable relationships through graph learning, attention, or channel mixing. Selective approaches such as DUET (Qiu et al., 2025) seek to retain informative interactions while limiting irrelevant ones. A related line of work considers directional predictive relationships: Granger causality defines a classical predictive criterion (Granger, 1969), neural extensions model nonlinear dependencies (Tank et al., 2022; Cheng et al., 2022), and Causalformer (Zhang et al., 2023), CAIFormer (Zhang et al., 2026), and ARCausal (Zhou et al., 2026) use causal graphs or masks in forecasting. GRAIL instead uses multi-step lagged predictive evidence to construct an input-dependent sparse mask for cross-channel aggregation.

3 METHODOLOGY

3.1 PROBLEM DEFINITION

Given a historical multivariate time series $\mathbf{X} \in \mathbb{R}^{B\times L\times N}$, where B, L, and N denote the batch size, look-back length, and number of variables, respectively, MTSF aims to predict the future sequence $\hat{\mathbf{Y}} \in \mathbb{R}^{B\times H\times N}$ over the prediction horizon H:

$$\hat{\mathbf{Y}} = f_\theta(\mathbf{X}). \tag{1}$$

Real-world multivariate series often exhibit heterogeneous temporal patterns and delayed cross-channel dependencies. The former requires adaptive temporal modeling across different temporal scales, while the latter requires identifying informative interactions among variables beyond simple correlation-based aggregation.

3.2 OVERALL ARCHITECTURE

Figure 2 illustrates the overall architecture of GRAIL. The framework consists of two complementary modules: TSF for adaptive temporal representation learning and GICM for sparse predictive dependency modeling. We apply RevIN Kim et al. (2021) before temporal modeling to alleviate distribution shifts, while estimating the predictive dependency mask from the raw input to preserve the original amplitude and statistical information.

The overall process is formulated as:

$$X^{norm} = \text{RevIN}(X) \tag{2}$$

$$X^{temp} = \text{TSF}(X^{norm}) \tag{3}$$

$$\mathcal{M} = \text{GICM}(X) \tag{4}$$

$$X^{mix} = \text{MMA}(X^{temp}, \mathcal{M}), \hat{Y} = \text{Predictor}(X^{mix}), \tag{5}$$

where $X^{temp} \in \mathbb{R}^{B\times N\times D}$ denotes the temporal representation, $\mathcal{M} \in \mathbb{R}^{B\times N\times N}$ denotes the sparse predictive dependency mask, and MMA denotes masked multivariate attention. TSF and GICM are computed in parallel and integrated through lag-aware cross-channel aggregation.

3.3 TEMPORAL SCALE FUSION

The TSF module is designed to capture sample-dependent temporal heterogeneity. Rather than using a single temporal receptive field, TSF constructs three branches with short, base, and long scales to model local fluctuations, periodic variations, and long-term trends, respectively.

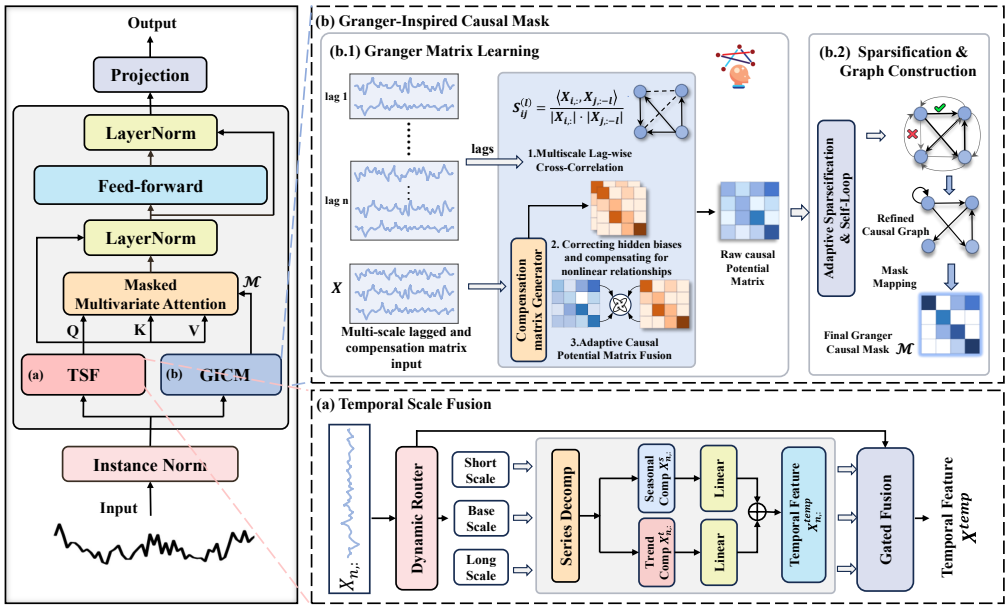


Figure 2: Overall framework of GRAIL. (a) TSF adaptively fuses multi-scale temporal representations. (b) GICM constructs sparse lagged predictive masks: (b.1) estimates multi-step Granger-inspired dependencies, and (b.2) sparsifies them into the final mask for structure-aware channel aggregation.

For the n-th variable, each branch $b \in \{short, base, long\}$ decomposes the normalized sequence $\mathbf{x}_n^{norm} \in \mathbb{R}^L$ into trend and residual components:

$$X_{n,:}^{trend,b} = \text{AvgPool}_{k_b}(\text{Padding}(X_{n,:}^{norm})), \tag{6}$$

$$X_{n,:}^{seas,b} = X_{n,:}^{norm} - X_{n,:}^{trend,b}, \tag{7}$$

The two components are projected into a shared hidden space:

$$\mathbf{z}_n^b = X_{n,:}^{trend,b} \cdot W_t^b + X_{n,:}^{seas,b} \cdot W_s^b, \tag{8}$$

where $W_t^b, W_s^b \in \mathbb{R}^{L \times D}$ are learnable projection matrices, and $\mathbf{z}_n^b \in \mathbb{R}^D$ denotes the branch-specific representation.

To adaptively combine different temporal scales, TSF introduces a lightweight sample-level router. We summarize the normalized input using three global statistics:

$$s_1 = \text{mean}_{t,n}(X^{\text{norm}}) \tag{9}$$

$$s_2 = \text{std}_{t,n}(X^{\text{norm}}) \tag{10}$$

$$s_3 = \text{mean}_{t,n}(|\Delta X^{\text{norm}}|), \tag{11}$$

which are concatenated as $\mathbf{s} = [s_1, s_2, s_3]$. The scale weights are generated by:

$$\mathbf{w} = \text{Softmax}\left(\text{MLP}(\mathbf{s})\right), \tag{12}$$

where $\mathbf{w} = [w_{short}, w_{base}, w_{long}]$. The final temporal representation of variable n is:

$$\mathbf{z}_n = \sum_{b \in \{short, base, long\}} w_b \mathbf{z}_n^b, \tag{13}$$

All variable representations are stacked into $\mathbf{Z} \in \mathbb{R}^{B \times N \times D}$. To prevent the router from assigning overly concentrated weights, we use a balance regularization:

$$\mathcal{L}_{bal} = \sum_{b \in \{short, base, long\}} \left(\mathbb{E}_{batch}[w_b] - \frac{1}{3} \right)^2, \tag{14}$$

3.4 GRANGER-INSPIRED CAUSAL MASK

GICM constructs a sparse and directional predictive dependency prior among variables by leveraging multi-step lag information. Inspired by the predictive perspective of Granger causality, GICM evaluates whether the historical information of a source variable j provides useful signals for forecasting a target variable i. The resulting mask serves as a forecasting-oriented structural prior to guide cross-channel aggregation, rather than attempting to recover ground-truth interventional causality.

For each lag step ℓ, the lag-aware predictive score from variable j to variable i is computed as:

$$S_{ij}^{(\ell)} = \left[\frac{\langle \mathbf{X}_{i,\ell+1:L}, \mathbf{X}_{j,1:L-\ell} \rangle}{\|\mathbf{X}_{i,\ell+1:L}\|_2 \, \|\mathbf{X}_{j,1:L-\ell}\|_2} \right]_+, \tag{15}$$

where $[\cdot]_+ = \max(\cdot, 0)$. Since $S_{ij}^{(\ell)}$ and $S_{ji}^{(\ell)}$ correspond to opposite temporal directions, the resulting score naturally captures asymmetric dependency patterns.

To account for diverse response delays across variables, GICM aggregates predictive scores from K lag steps:

$$G_{ij} = \sum_{\ell=1}^{K} \pi_\ell S_{ij}^{(\ell)}, \tag{16}$$

$$\pi_\ell = \frac{\exp(\omega_\ell)}{\sum_{m=1}^{K} \exp(\omega_m)}, \tag{17}$$

where ω_ℓ is a learnable lag-importance parameter. The resulting matrix $G \in \mathbb{R}^{B \times N \times N}$ represents the initial lag-aware predictive dependency prior.

Although lag information provides an efficient dependency estimation mechanism, fixed pairwise scores may not fully capture sample-specific variations. Therefore, GICM introduces an adaptive residual matrix $\mathbf{R}$ to provide learnable corrections:

$$\widetilde{G} = \alpha G + (1 - \alpha) R, \quad \alpha = \sigma(a), \tag{18}$$

where a is a learnable scalar. This formulation preserves the explicit lag-aware dependency prior while allowing flexible adaptation to different forecasting scenarios.

To obtain a compact dependency representation, we normalize and sparsify $\widetilde{\mathbf{G}}$. First, row-wise normalization is applied:

$$\mathbf{P} = \mathrm{RowSoftmax}\left(\widetilde{\mathbf{G}}\right), \tag{19}$$

For each target variable i, the adaptive threshold is:

$$\tau_i = \rho \cdot \mathrm{sg}\left(\frac{1}{N-1} \sum_{j \neq i} P_{ij} \right), \tag{20}$$

where ρ is a learnable sparsification coefficient and $\mathrm{sg}(\cdot)$ denotes the stop-gradient operation. The sparse dependency mask is then generated by:

$$\overline{\mathcal{M}}_{ij} = P_{ij} \cdot \mathrm{Gate}\left(P_{ij}, \tau_i\right), \quad i \neq j, \tag{21}$$

where $\overline{\mathcal{M}}$ denotes the sparsified cross-channel dependency mask. To retain each variable's own temporal information, we further introduce adaptive self-loops:

$$\mathcal{M} = \overline{\mathcal{M}} + \beta \mathbf{I}, \beta = \sigma(\theta_{\mathrm{loop}}), \tag{22}$$

where $\mathbf{I}$ is the identity matrix and β controls the contribution of self-dependency. The final mask $\mathcal{M}$ therefore provides sparse, asymmetric, and lag-aware dependency guidance while preserving each variable's own information.

3.5 Masked Multivariate Attention and Prediction

Given the temporal representation $\mathbf{X}^{\text{temp}} \in \mathbb{R}^{B\times N\times D}$ and the final dependency mask $\mathcal{M} \in \mathbb{R}^{B\times N\times N}$, MMA performs mask-guided attention across variables. Specifically,

$$\mathbf{Q} = \mathbf{X}^{\text{temp}} W_Q, \mathbf{K} = \mathbf{X}^{\text{temp}} W_K, \mathbf{V} = \mathbf{X}^{\text{temp}} W_V, \tag{23}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{D\times D}$. The dependency mask is incorporated into the channel-wise attention logits:

$$\mathbf{A} = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D}} + \log(\mathcal{M} + \epsilon)\right), \tag{24}$$

where ϵ is a small constant for numerical stability. The aggregated representation is

$$\mathbf{X}^{\text{mix}} = \mathbf{A}\mathbf{V}. \tag{25}$$

This formulation guides cross-variable aggregation toward source variables with stronger lag-aware predictive dependencies.

The final prediction is produced by a forecasting head followed by the inverse RevIN transformation:

$$\hat{\mathbf{Y}} = \text{RevIN}^{-1}\left(\text{Head}\left(\mathbf{X}^{\text{mix}}\right)\right). \tag{26}$$

The training objective combines forecasting loss with balance regularization:

$$\mathcal{L} = \mathcal{L}_{\text{pred}}(\hat{\mathbf{Y}}, \mathbf{Y}) + \lambda_{\text{bal}}\mathcal{L}_{\text{bal}}, \tag{27}$$

where $\mathcal{L}_{\text{pred}}$ is the forecasting loss and λ_{bal} controls balance regularization. Training and checkpoint-selection settings are detailed in Appendix A.3.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate GRAIL following the Time Series Forecasting Benchmark (TFB) protocol (Qiu et al., 2024). Experiments are conducted on eight widely used real-world multivariate forecasting datasets from diverse domains, including ETTh1, ETTh2, ETTm1, ETTm2, Weather, Traffic, Electricity, and Solar-Energy. Dataset statistics and splits appear in Appendix A.1; runtime, optimization, and model settings are detailed in Appendices A.3–A.4.

Baselines. We compare GRAIL with representative MTSF models from different modeling paradigms. The compared methods include linear forecasting models (DLinear (Zeng et al., 2023)), channel-independent patch-based models (PatchTST (Nie et al., 2022)), temporal representation learning models (TimesNet (Wu et al., 2022) and MSGNet (Cai et al., 2024)), and channel-dependent Transformer-based models (Crossformer (Zhang & Yan, 2022) and iTransformer (Liu et al., 2024)). We further include DUET (Qiu et al., 2025), which jointly considers temporal and channel dependencies, HASPFormer (Ljubić et al., 2026), a recent forecasting model, and ARCausal (Zhou et al., 2026), which adopts causal-aware dependency modeling.

4.2 Main Results

Table 1 reports forecasting results on eight benchmark datasets. Across 32 dataset–horizon settings, GRAIL achieves the best MSE in 16 settings and the best MAE in 25, showing competitive or superior performance across the evaluated tasks. These results support the value of combining adaptive temporal modeling with lag-aware cross-channel aggregation.

Compared with channel-independent methods such as DLinear and PatchTST, GRAIL performs particularly well on Traffic, Electricity, and Solar-Energy, where cross-variable information may be useful. It also achieves strong results relative to channel interaction models such as Crossformer, iTransformer, and DUET. The comparison suggests that selective, lag-aware aggregation is a promising way to use cross-channel information without relying on dense interactions.

GRAIL outperforms ARCausal in many forecasting settings while remaining competitive where ARCausal performs strongly. ARCausal constructs dependency masks using transfer-entropy and

Table 1: Multivariate forecasting results across prediction horizons $\{96, 192, 336, 720\}$ with input length 96. The best and second-best results are highlighted in bold and underlined, respectively.

Models		GRAIL (ours)		ARCausal (2026)		HASPFormer (2026)		DUET (2025)		iTransformer (2024)		MSGNet (2024)		PatchTST (2023)		Crossformer (2023)		TimesNet (2023)		DLinear (2023)	
Metrics		mse	mae	mse	mae	mse	mae	mse	mae	mse	mae	mse	mae	mse	mae	mse	mae	mse	mae	mse	mae
ETTh1	96	0.376	**0.392**	**0.372**	0.393	0.377	0.399	0.377	0.393	0.386	0.405	0.390	0.411	0.414	0.419	0.423	0.448	0.384	0.402	0.386	0.400
	192	0.431	**0.423**	**0.427**	0.426	0.430	0.428	0.429	0.425	0.441	0.436	0.442	0.442	0.460	0.445	0.471	0.474	0.436	0.429	0.437	0.432
	336	**0.471**	**0.444**	0.474	0.449	0.474	0.450	**0.471**	0.446	0.487	0.458	0.480	0.468	0.501	0.466	0.570	0.546	0.491	0.469	0.481	0.459
	720	**0.468**	**0.466**	0.475	0.467	0.484	0.475	0.496	0.480	0.503	0.491	0.494	0.488	0.500	0.488	0.653	0.621	0.521	0.500	0.519	0.516
ETTh2	96	**0.288**	**0.337**	0.295	0.345	0.296	0.346	0.296	0.345	0.297	0.349	0.328	0.371	0.302	0.348	0.745	0.584	0.340	0.374	0.333	0.387
	192	0.370	0.390	0.375	0.394	0.374	0.396	**0.368**	**0.389**	0.380	0.400	0.402	0.414	0.388	0.400	0.877	0.656	0.402	0.414	0.477	0.476
	336	**0.410**	**0.422**	0.419	0.429	0.421	0.432	0.411	**0.422**	0.428	0.432	0.435	0.443	0.426	0.433	1.043	0.731	0.452	0.452	0.594	0.541
	720	0.421	0.437	0.426	0.443	0.431	0.448	**0.412**	**0.434**	0.427	0.445	0.417	0.441	0.431	0.446	1.104	0.763	0.462	0.468	0.831	0.657
ETTm1	96	0.324	0.353	**0.316**	**0.351**	0.317	0.355	0.324	0.354	0.334	0.368	0.319	0.366	0.329	0.367	0.404	0.426	0.338	0.375	0.345	0.372
	192	0.368	0.379	**0.360**	**0.374**	0.366	0.384	0.369	0.379	0.377	0.391	0.376	0.397	0.367	0.385	0.450	0.451	0.374	0.387	0.380	0.389
	336	0.403	0.402	**0.397**	**0.399**	0.405	0.410	0.404	0.402	0.426	0.420	0.417	0.422	0.399	0.410	0.532	0.515	0.410	0.411	0.413	0.413
	720	0.466	0.438	0.467	0.440	0.463	0.445	0.463	**0.437**	0.491	0.459	0.481	0.458	**0.454**	0.439	0.666	0.589	0.478	0.450	0.474	0.453
ETTm2	96	**0.170**	**0.250**	0.177	0.259	0.176	0.259	0.174	0.255	0.180	0.264	0.177	0.262	0.175	0.259	0.287	0.366	0.187	0.267	0.193	0.292
	192	**0.234**	**0.291**	0.238	0.298	0.241	0.301	0.243	0.302	0.250	0.309	0.247	0.307	0.241	0.302	0.414	0.492	0.249	0.309	0.284	0.362
	336	0.300	**0.334**	**0.296**	0.335	0.301	0.340	0.304	0.341	0.311	0.348	0.312	0.346	0.305	0.343	0.597	0.542	0.321	0.351	0.369	0.427
	720	0.408	**0.396**	0.405	0.398	0.401	0.399	**0.399**	0.397	0.412	0.407	0.414	0.403	0.402	0.400	1.730	1.042	0.408	0.403	0.554	0.522
Weather	96	**0.156**	**0.193**	0.169	0.208	0.164	0.209	0.163	0.202	0.174	0.214	0.163	0.212	0.177	0.218	0.158	0.230	0.172	0.220	0.196	0.255
	192	**0.204**	**0.240**	0.211	0.250	0.210	0.251	0.218	0.252	0.221	0.254	0.212	0.254	0.225	0.259	0.206	0.277	0.219	0.261	0.237	0.296
	336	**0.262**	**0.284**	0.273	0.292	0.268	0.293	0.274	0.294	0.278	0.296	0.272	0.299	0.278	0.297	0.272	0.335	0.280	0.306	0.283	0.335
	720	**0.330**	**0.331**	0.345	0.344	0.349	0.346	0.349	0.343	0.358	0.347	0.350	0.348	0.354	0.348	0.398	0.418	0.365	0.351	0.345	0.381
Electricity	96	0.147	0.238	0.145	0.234	**0.138**	**0.231**	0.145	0.233	0.148	0.240	0.165	0.274	0.181	0.281	0.219	0.314	0.168	0.272	0.197	0.282
	192	**0.159**	**0.245**	0.160	0.247	**0.159**	0.251	0.163	0.248	0.162	0.253	0.184	0.292	0.188	0.274	0.231	0.322	0.184	0.289	0.196	0.285
	336	**0.171**	**0.260**	0.173	0.263	0.173	0.267	0.175	0.262	0.178	0.269	0.195	0.302	0.204	0.293	0.246	0.337	0.198	0.300	0.209	0.301
	720	0.201	**0.284**	0.210	0.297	**0.196**	0.290	0.204	0.291	0.225	0.317	0.231	0.332	0.246	0.324	0.280	0.363	0.220	0.330	0.245	0.333
Solar	96	**0.191**	**0.202**	0.195	0.207	0.195	0.229	0.200	0.207	0.203	0.237	0.231	0.271	0.234	0.286	0.310	0.331	0.250	0.292	0.290	0.378
	192	**0.225**	**0.222**	0.229	0.229	0.227	0.255	0.228	0.233	0.233	0.261	0.265	0.286	0.267	0.310	0.734	0.725	0.296	0.318	0.320	0.398
	336	**0.241**	**0.235**	0.253	0.247	**0.241**	0.269	0.262	0.244	0.248	0.273	0.283	0.300	0.290	0.315	0.750	0.735	0.319	0.330	0.353	0.415
	720	**0.241**	**0.240**	0.268	0.257	0.243	0.272	0.258	0.249	0.249	0.275	0.295	0.313	0.289	0.317	0.769	0.765	0.338	0.337	0.356	0.413
Traffic	96	0.402	**0.249**	**0.389**	0.256	0.410	0.252	0.407	0.252	0.395	0.268	0.600	0.331	0.462	0.295	0.522	0.290	0.593	0.321	0.650	0.396
	192	0.422	**0.260**	**0.408**	0.264	0.429	0.268	0.431	0.262	0.417	0.276	0.620	0.344	0.466	0.296	0.530	0.293	0.617	0.336	0.598	0.370
	336	0.435	**0.266**	**0.424**	0.272	0.445	0.278	0.456	0.269	0.433	0.283	0.641	0.354	0.482	0.304	0.558	0.305	0.629	0.336	0.605	0.373
	720	0.464	**0.285**	**0.459**	0.290	0.508	0.295	0.509	0.292	0.467	0.302	0.687	0.386	0.514	0.322	0.589	0.328	0.640	0.350	0.645	0.394
1^{st} Count		**16**	**25**	10	3	4	1	4	4	0	0	0	0	1	0	0	0	0	0	0	0

Table 2: Ablation of GRAIL components and dependency masks. Remove disables or simplifies components; Replace uses alternative masks. Results are averaged over forecasting horizons. Lower is better; the best value for each dataset and metric is shown in **bold**.

	Variant	ETTh2		Weather		Solar-Energy		Traffic	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
	GRAIL	**0.372**	**0.396**	**0.238**	**0.262**	**0.225**	**0.224**	**0.431**	**0.265**
Remove	w/o TSF	0.379	0.402	0.241	0.264	0.226	0.232	0.442	0.267
Remove	w/o GICM	0.382	0.403	0.247	0.267	0.339	0.303	0.650	0.350
Remove	w/o Mask	0.378	0.401	0.246	0.266	0.249	0.255	0.441	0.266
Remove	w/o Lag	0.375	0.399	0.241	0.264	0.241	0.242	0.444	0.266
Replace	Euclid	0.378	0.401	0.247	0.267	0.226	0.229	0.447	0.267
Replace	DTW	0.375	0.400	0.246	0.267	0.232	0.231	0.445	0.266
Replace	Mahalanobis	0.378	0.400	0.246	0.267	0.235	0.230	0.446	0.267
Replace	Random Mask	0.375	0.399	0.244	0.265	0.241	0.232	0.440	0.266

VAR-based estimation, whereas GRAIL estimates multi-step lagged predictive dependencies from historical observations and combines them with adaptive temporal-scale representations. The results indicate that this combination is effective under the evaluated forecasting protocol.

Performance patterns differ across datasets: GRAIL is strong on ETT and Weather as well as on datasets with more complex cross-variable interactions, including Traffic, Electricity, and Solar-Energy. The ablation study examines the respective contributions of TSF and GICM to these results.

4.3 Ablation Study

We organize the variants into two groups. Remove includes w/o TSF, w/o GICM, w/o Mask (full channel attention), and w/o Lag (single-lag dependency estimation). Replace substitutes the proposed mask with Euclidean, DTW, Mahalanobis, or random masks while retaining the remaining architecture. Table 2 reports results on four datasets, averaged over forecasting horizons. GRAIL achieves the lowest MSE and MAE in all eight dataset–metric comparisons. Removing TSF modestly increases error across the datasets, while removing GICM has a larger effect on Solar-Energy and Traffic: their MSE rises from 0.225 to 0.339 and from 0.431 to 0.650, respectively.

Replacing the sparse mask with full attention or using only one lag increases error on all four datasets. The Euclidean, DTW, Mahalanobis, and random masks also perform worse than GRAIL, supporting the use of multi-step lagged predictive information for channel selection. The Solar-Energy MSE margin over Euclid is small (0.225 versus 0.226), indicating dataset-dependent benefits. Appendix B.2 reports additional controls under the supplementary MSE protocol.

4.4 Qualitative Analysis of Learned Channel Weights

Figure 3 examines whether the selected lagged relationships are reflected in the historical sequences and whether the resulting channel weights are selective. In panel (a), each source sequence is shifted by its estimated predictive lag before comparison with the target. The displayed HUFL→LUFL, LULL→MULL, and LUFL→HUFL pairs show locally aligned fluctuations after shifting. This alignment is consistent with the lagged predictive associations used by GICM, although the visualization alone does not establish a causal relationship. Panel (b) shows that the masked weights are unevenly distributed across source channels. Several entries receive little weight, while a smaller set receives larger weights; the pattern also differs across target directions. Together, the panels illustrate how lag-aware selection guides cross-channel aggregation on ETTh2. Additional forecasting and channel visualizations appear in Appendix C.

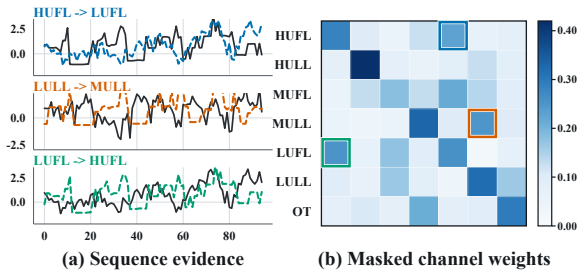


Figure 3: Selected lag-aligned channel pairs and learned masked channel weights on ETTh2.

4.5 Parameter Sensitivity with Varying Lag Steps

Table 3 compares four lag-step counts, $K \in \{3, 6, 9, 12\}$, on ETTh1, Weather, Electricity, and Traffic at prediction horizon $H = 96$. With $K = 3$, GRAIL obtains the lowest MSE and MAE on ETTh1 and Traffic and ties for the best results on Weather. Electricity is the one exception: $K = 6$ improves MSE by 0.001 relative to $K = 3$, while $K = 3$ gives the lowest MAE. Increasing the lag range therefore yields no consistent gain at this horizon. Traffic shows the clearest change: its MSE rises from 0.402 at $K = 3$ to 0.431 at $K = 12$, and its MAE rises from 0.249 to 0.260. Weather changes little across the tested values. These results suggest that three lag steps provide sufficient dependency coverage for these settings. However, this comparison is restricted to $H = 96$, and the preferred lag range may change with the forecasting horizon. The different responses across datasets also suggest that longer lag coverage should be assessed for each setting, rather than assumed to improve prediction uniformly. We use $K = 3$ as the default to limit the number of lag evaluations while retaining competitive accuracy; additional lag and context studies appear in Appendix B.3.

Table 3: Lag-step sensitivity at $H = 96$; entries report MSE/MAE.

K	ETTh1	Weather	Electricity	Traffic
3	**0.376/0.392**	**0.156/0.193**	0.148/**0.237**	**0.402/0.249**
6	0.379/0.394	0.157/0.195	**0.147**/0.238	0.411/0.254
9	0.378/0.393	**0.156/0.193**	0.152/0.241	0.421/0.258
12	0.381/0.395	**0.156/0.193**	0.148/0.238	0.431/0.260

4.6 Interpretability Case Study

Figure 4 examines how GICM uses historical source variables when forecasting a target channel in Electricity. Panel (a) compares the target with the selected sources over the same input window. Their trajectories share several rises and falls, although the match is not uniform throughout the window. Panel (b) shows that a small set of source channels receives the largest non-self masked attention shares, while the remaining channels receive smaller shares. Panel (c) adds the time direction: the selected sources remain correlated with the target when they precede it by several hours, and the correlations weaken at longer leads. Together, the three views suggest that GICM concentrates on a few sources whose histories contain temporally aligned information. This case illustrates predictive associations learned by the model; it does not establish interventional causal effects.

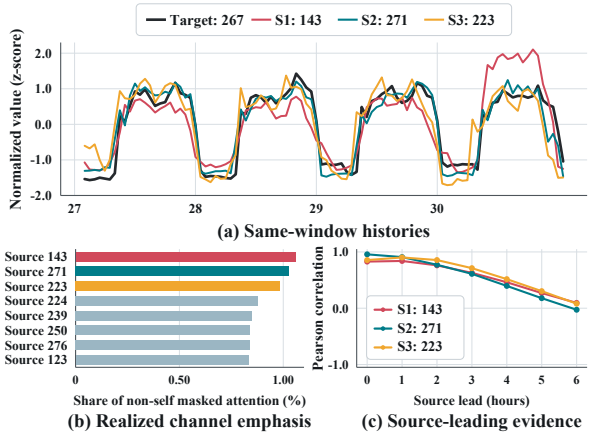


Figure 4: Interpretability case study on Electricity. GICM identifies source variables with similar historical patterns and source-leading lagged correlations.

4.7 Model Efficiency

Figure 5 compares forecasting error, training time per iteration, and memory use on ETTm1 with 96 input and 96 prediction steps. The horizontal axis reports training time, the vertical axis reports MSE, and bubble size encodes memory. GRAIL lies near the low-error, short-time region of the plot. Its error is lower than DUET's at a similar training time, while Crossformer, TimesNet, and MSGNet take longer per iteration in this experiment. GRAIL also avoids the largest memory bubbles among the plotted methods. These observations show a favorable balance between forecasting accuracy and resource use in this setting. Runtime and memory depend on implementation and hardware, so the plot supports a comparison under the reported configuration rather than a general efficiency ranking. Appendix D reports additional profiling of internal controls.

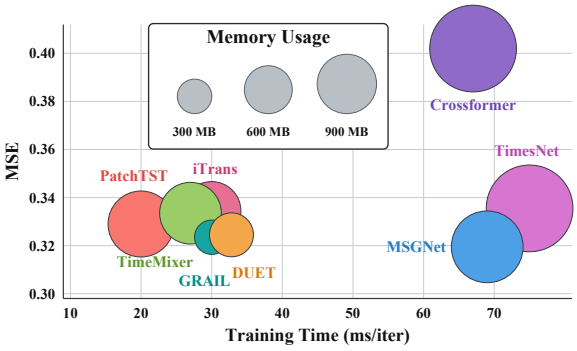


Figure 5: Efficiency comparison on ETTm1 under the 96-input/96-output setting. Bubble size represents memory usage.

5 Conclusion

We propose GRAIL, a multivariate forecasting framework that jointly models temporal heterogeneity and lag-aware predictive channel dependencies. GRAIL combines Temporal Scale Fusion for adaptive multi-scale temporal representation learning with Granger-Inspired Causal Masking for sparse dependency-guided channel aggregation. Experiments on eight real-world benchmarks, together with ablation and qualitative analyses, demonstrate its effectiveness. The results highlight the benefit of integrating adaptive temporal modeling with sparse lag-aware dependency guidance for multivariate time series forecasting.

Reproducibility Statement

Dataset statistics and preprocessing procedures are provided in Appendix A.1. The computing environment, optimization settings, and detailed model hyperparameters are documented in Appendices A.3 and A.4. Definitions of the supplementary controls and their evaluation settings are given in Appendix A, with additional results in Appendix B. The supplementary training protocol is explicitly distinguished from the main experimental protocol to clarify the conditions under which the results were obtained.

AI Use Statement

Generative AI tools were used to improve the clarity and readability of author-provided text and to assist with LaTeX formatting. The authors reviewed the suggested edits and take responsibility for the final manuscript.

References

Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. MSGNet: Learning multi-scale inter-series correlations for multivariate time series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10):11141–11149, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i10.28991.

Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *The Twelfth International Conference on Learning Representations*, October 2023.

Yuxiao Cheng, Runzhao Yang, Tingxiong Xiao, Zongren Li, Jinli Suo, Kunlun He, and Qionghai Dai. CUTS: Neural causal discovery from irregular time-series data. In *The Eleventh International Conference on Learning Representations*, September 2022.

C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. *Advances in Neural Information Processing Systems*, 37:64145–64175, December 2024. doi: 10.52202/079017-2046.

Xuanwen Huang, Yang Yang, Yang Wang, Chunping Wang, Zhisheng Zhang, Jiarong Xu, Lei Chen, and Michalis Vazirgiannis. DGraph: A large-scale financial dataset for graph anomaly detection. *Advances in Neural Information Processing Systems*, 35:22765–22777, December 2022.

Duc Kieu, Tung Kieu, Peng Han, Bin Yang, Christian S. Jensen, and Bac Le. TEAM: Topological evolution-aware framework for traffic forecasting–extended version. *Proceedings of the VLDB Endowment*, 18(2):265–278, October 2024. ISSN 2150-8097. doi: 10.14778/3705829.3705844.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, October 2021.

Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-Tao Xia. TimeBridge: Non-stationarity matters for long-term time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 39815–39840. PMLR, October 2025.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. *International Conference on Learning Representations*, 2024:11116–11140, May 2024.

Hrvoje Ljubić, Goran Martinović, Tomislav Volarić, and Daniel Vasić. HASPFormer: Advancing multivariate time-series forecasting with self-attention and stochastic pooling. *Advanced Engineering Informatics*, 71:104212, April 2026. ISSN 1474-0346. doi: 10.1016/j.aei.2025.104212.

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*, September 2022.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. TFB: Towards comprehensive and fair benchmarking of time series forecasting methods. *Proceedings of the VLDB Endowment*, 17 (9):2363–2377, May 2024. ISSN 2150-8097. doi: 10.14778/3665844.3665863.

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 V.1*, KDD '25, pp. 1185–1196, New York, NY, USA, July 2025. Association for Computing Machinery. ISBN 979-8-4007-1245-6. doi: 10.1145/3690624.3709325.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning : A systematic literature review: 2005–2019. *Applied Soft Computing*, 90:106181, May 2020. ISSN 1568-4946. doi: 10.1016/j.asoc.2020.106181.

Yanru Sun, Zongxia Xie, Haocheng Wang, Xin Huang, and Qinghua Hu. Solar wind speed prediction via graph attention network. *Space Weather*, 20(7):e2022SW003128, 2022. ISSN 1542-7390. doi: 10.1029/2022SW003128.

Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily B. Fox. Neural granger causality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(8):4267–4279, August 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3065601.

Hao Wang, Zhiyu Wang, Yunlong Niu, Zhaoran Liu, Haozhe Li, Yilin Liao, Yuxin Huang, and Xinggao Liu. An accurate and interpretable framework for trustworthy process monitoring. *IEEE Transactions on Artificial Intelligence*, 5(5):2241–2252, May 2024a. ISSN 2691-4581. doi: 10.1109/TAI.2023.3319606.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. *International Conference on Learning Representations*, 2024:38626–38652, May 2024b.

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.

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*, September 2022.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 753–763, New York, NY, USA, August 2020. Association for Computing Machinery. ISBN 978-1-4503-7998-4. doi: 10.1145/3394486.3403118.

Sean Bin Yang, Chenjuan Guo, and Bin Yang. Context-aware path ranking in road networks. *IEEE Transactions on Knowledge and Data Engineering*, 34(7):3153–3168, July 2022. ISSN 1558-2191. doi: 10.1109/TKDE.2020.3025024.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):11121–11128, June 2023. ISSN 2374-3468. doi: 10.1609/aaai.v37i9.26317.

Chonghao Zhang, Linlin Zhao, Zhefeng Yin, and Zhenguo Zhang. Causalformer: Causal discovery-based transformer for multivariate time series forecasting. In *2023 16th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1–6. IEEE, 2023.

Xingyu Zhang, Hanyun Du, Zeen Song, Siyu Zhao, Changwen Zheng, and Wenwen Qiang. Beyond all-to-all: Causal-aligned transformer with dynamic structure learning for multivariate time series forecasting, February 2026.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, September 2022.

Chengli Zhou, Zicheng Wang, Yaqun Huang, Dapeng Tao, and Chunna Zhao. Causal mask in transformer via transfer entropy estimation from vector autoregressive learning for multivariate time series forecasting. *International Journal of Neural Systems*, 36(08):2650019, 2026.

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*, pp. 27268–27286. PMLR, June 2022.

A Experimental Setup

This appendix extends the main-text analysis of TSF and GICM across eight datasets, with component studies on ETTh2 and Weather. Supplementary runs use the MSE objective, seed 2021, and the settings below.

A.1 Datasets and Evaluation

Table A1 summarizes the data used in these runs. The ETT subsets contain seven transformer-related variables at hourly or 15-minute resolution. Weather contains 21 meteorological variables sampled every 10 minutes. Electricity describes hourly consumption across 321 channels, Solar-Energy records 137 solar-generation channels every 10 minutes, and Traffic contains hourly measurements from 862 channels. The reported lengths are the effective lengths read by the experiment pipeline.

Table A1: Dataset statistics and chronological split sizes. N is the number of variables; lengths count timestamps.

Dataset	N	Length	Train	Validation	Test	Interval
ETTh1	7	14,400	8,640	2,880	2,880	1 hour
ETTh2	7	14,400	8,640	2,880	2,880	1 hour
ETTm1	7	57,600	34,560	11,520	11,520	15 min
ETTm2	7	57,600	34,560	11,520	11,520	15 min
Weather	21	52,696	36,886	5,270	10,540	10 min
Electricity	321	26,304	18,412	2,631	5,261	1 hour
Solar-Energy	137	52,560	31,536	10,512	10,512	10 min
Traffic	862	17,544	12,280	1,755	3,509	1 hour

Chronological train/validation/test ratios are 60/20/20 for ETT and Solar-Energy, and 70/10/20 for Weather, Electricity, and Traffic, up to rounding. Channel-wise standardization is fitted exclusively to training observations. Historical inputs may precede a validation or test boundary, while forecast targets remain within their assigned split. Evaluation uses every valid forecast origin with stride one. MSE and MAE are averaged over test windows, forecast steps, and channels in the training-standardized scale.

A.2 Training Configuration

The supplementary base runs use $L = 96$, $K = 3$, batch size 32, and $H \in \{96, 192, 336, 720\}$. Table A2 lists settings shared across horizons; Table A5 lists temporal windows.

Table A2: Dataset-specific base settings. D denotes model dimension; epochs are maximum training budgets.

Dataset	D	Learning rate	Maximum epochs
ETTh1	512	0.0005	100
ETTh2	512	0.0005	100
ETTm1	512	0.0005	100
ETTm2	128	0.0005	100
Weather	512	0.0010	100
Electricity	512	0.0005	100
Solar-Energy	512	0.0010	100
Traffic	512	0.0005	10

For the supplementary runs, the forecasting term in the main-text objective is MSE. Training uses Adam and the implemented TSF balance regularizer. Gradients are clipped at norm 1.0. Early stopping may terminate training before the maximum budget, and the checkpoint with the lowest validation MSE is used for testing.

A.3 Computing Environment and Optimization

Main-experiment environment. The original experimental record specifies an NVIDIA vGPU with 32 GB of allocated GPU memory. The software stack is Ubuntu 20.04, Python 3.8, PyTorch 2.0.0, and CUDA 11.8. Experiments are launched on CUDA device with one TFB benchmark worker. The record identifies a virtual GPU allocation but does not specify the underlying physical GPU model or server CPU.

Table A3: Recorded environment of the original main experiments. These specifications are not independently verified for the additional supplementary runs.

Item	Recorded configuration
GPU	NVIDIA vGPU, 32 GB allocated memory
Operating system	Ubuntu 20.04
Python	3.8
PyTorch	2.0.0
CUDA	11.8
Execution	CUDA device ; one TFB worker
Seed	2021

Optimization and data loading. Adam uses $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and zero weight decay. Mini-batch size is 32, and gradients are clipped to norm 1.0. Training windows are shuffled each epoch; the final incomplete training batch is dropped, while incomplete validation and test batches are retained. Channel-wise training-set standardization is followed by the model's internal reversible instance normalization.

Learning-rate schedule. The initial rate η_0 follows Table A2. After epoch e, the implemented `type3` schedule sets the learning rate for the next epoch to

$$\eta_{\text{next}}(e) = \begin{cases} \eta_0, & e < 3, \\ \eta_0\, 0.9^{e-3}, & e \geq 3. \end{cases} \tag{28}$$

Early-stopping patience is five epochs for all datasets except Traffic, which uses three. Maximum training budgets are 100 and 10 epochs, respectively.

Training objective and evaluation metrics. The main experiments use MAE as the forecasting loss, together with TSF balance regularization, and select checkpoints by validation MAE. The supplementary experiments reported here use MSE as the forecasting loss and select checkpoints by validation MSE. Both report test MSE and MAE. Supplementary results therefore characterize the additional MSE-based protocol and are not direct replications of the main experiments. The supplementary default is $K = 3$, except for the single-lag control and lag-sensitivity runs; Table A5 lists the base settings.

Randomness and runtime scope. The original TFB record uses seed 2021 and its deterministic mode. Supplementary training and evaluation also use fixed seeds, but this does not quantify across-seed variability. The supplementary efficiency table specifies the timing procedure; the original hardware record alone does not establish that both sets of measurements used identical hardware.

A.4 Detailed Model Hyperparameters

Table A4 records the dataset-specific architecture settings supplied with the original experiment documentation. These settings are shared across prediction horizons within a dataset. d_{ff} is the feed-forward width, N_e the number of channel encoder layers, N_h the number of attention heads, p encoder dropout, p_h head dropout, and d_r router hidden width.

Table A4: Architecture settings from the original experimental record. The causal-correction hidden width is 64 throughout.

Dataset	D	d_{ff}	N_e	N_h	p	p_h	d_r
ETTh1	512	512	2	8	0.1	0.1	32
ETTh2	512	512	2	8	0.2	0.5	32
ETTm1	512	512	2	8	0.1	0.2	32
ETTm2	128	512	1	1	0.2	0.2	64
Weather	512	1024	2	8	0.1	0.2	32
Electricity	512	1024	2	1	0.5	0.0	32
Solar-Energy	512	1024	3	8	0.1	0.0	32
Traffic	512	1024	4	8	0.2	0.0	32

Temporal windows. Given base window parameter M, the short, base, and long branches use

$$M_s = \mathrm{odd}(\max(3, \lfloor M/2 \rfloor)), \tag{29}$$

$$M_b = \mathrm{odd}(M), \tag{30}$$

$$M_l = \mathrm{odd}(\max(M + 2, 2M - 1)), \tag{31}$$

where odd increases an even integer by one. Decomposition kernels are initialized as uniform moving averages and learned with softmax-normalized coefficients.

Table A5: Supplementary base settings by prediction horizon. All base runs use $L = 96$ and $K = 3$; sensitivity studies vary these explicitly.

	Base window M				
Dataset	96	192	336	720	Base K
ETTh1	13	25	31	31	3
ETTh2	13	25	31	35	3
ETTm1	12	20	31	31	3
ETTm2	13	25	25	31	3
Weather	25	25	35	35	3
Electricity	13	25	31	25	3
Solar-Energy	25	13	31	31	3
Traffic	13	31	25	31	3

As in the main text, K denotes the number of lag steps aggregated by GICM. The base model uses steps 1 through 3 with learned normalized weights. The single-lag control uses $K = 1$, and sensitivity runs additionally test $K = 6, 9, 12$. Checkpoints are selected on the chronological validation split.

A.5 Model Components and Controlled Variants

Following the main text, TSF integrates short-, base-, and long-scale temporal representations using input-dependent weights. GICM provides directional, lag-aware predictive guidance for masked multivariate attention (MMA). The forecasting head produces the requested horizon, followed by inverse RevIN. In the implementation used for the supplied plots, routing is evaluated for each window–channel pair; the routing visualization averages these weights over channels.

The supplementary controls examine TSF weighting and GICM dependency guidance. Dense attention corresponds to removing the predictive mask (w/o Mask in the main text), and single-lag estimation corresponds to removing multi-step lag aggregation (w/o Lag). Here, w/o Lag retains one lag step; it does not remove all lag information. The controls are defined as follows:

- **Base only** retains the base temporal branch.
- **Uniform fusion** retains all three branches with fixed weights of $1/3$.
- **Shared fusion** learns fusion logits that are independent of the input window.
- **Dense attention** retains channel aggregation without the predictive mask.
- **Single lag** restricts dependency estimation to $K = 1$.

The remaining dataset-specific settings are retained. Uniform and shared fusion preserve multi-scale representations, allowing their weighting mechanism to be compared with the learned input-dependent router. Base-only modeling also changes the representation capacity, so these controls are not assumed to have identical parameter counts.

A.6 Validation and Checkpoint Selection

Figure A1 shows validation MSE for the base models on ETTh2 and Weather at $H = 96$. ETTh2 exhibits early fluctuations, while Weather decreases after an initial peak. The trajectories are non-monotonic, supporting selection by validation performance rather than by the last completed epoch.

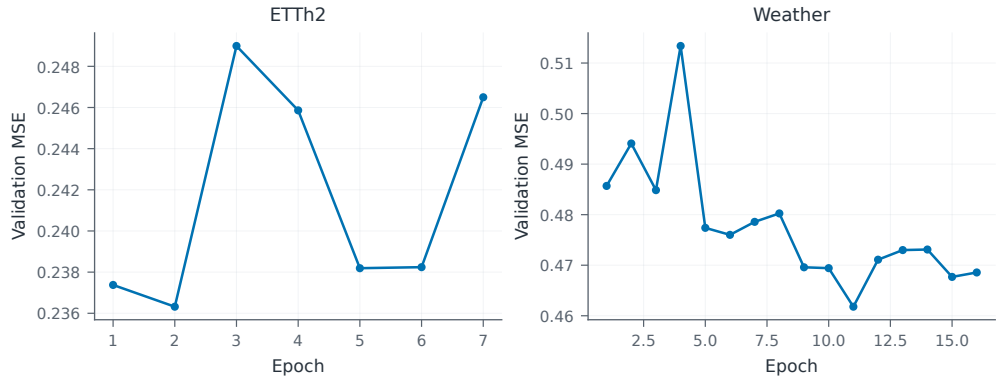


Figure A1: Validation MSE for GRAIL at $H = 96$. Curves end when training stops; test evaluation uses the validation-selected checkpoint.

The displayed trajectories describe one training seed. They do not quantify variation across independent initializations. The same checkpoint-selection rule is applied to the controlled variants and sensitivity runs, ensuring that all reported test metrics use a consistent selection procedure.

B Additional Quantitative Results

B.1 Forecast Horizon Dependence

Figure A2 summarizes the supplementary runs at four prediction horizons. MSE generally increases with the horizon. Solar-Energy is an exception: its MSE decreases from 0.3445 at $H = 336$ to 0.2983 at $H = 720$. These curves supplement the main-text comparison without repeating its complete result table.

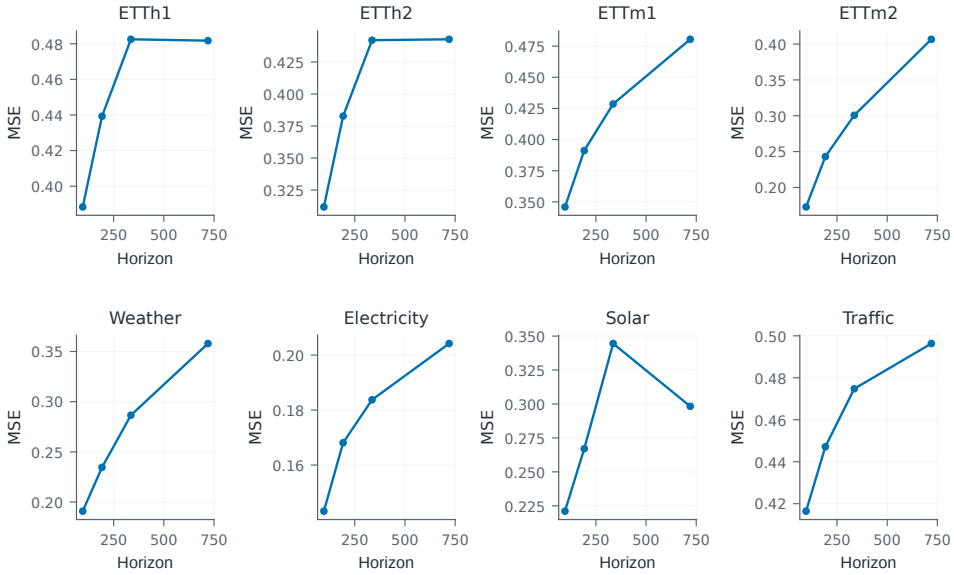


Figure A2: Forecasting MSE under the supplementary MSE-based protocol with seed 2021. Panels use independent vertical scales.

B.2 Temporal Fusion and Dependency Controls

Table A6 compares the controlled variants at $H = 96$. On ETTh2, all five controls achieve slightly lower MSE than GRAIL. On Weather, GRAIL improves on the three temporal-fusion controls, while dense attention and single-lag estimation achieve lower errors. These single-horizon, single-seed controls complement the horizon-averaged ablations in the main text. Under these supplementary settings, the gains are not uniform across datasets and variants.

Table A6: Supplementary controls at $H = 96$. Dense attention denotes w/o Mask; single lag denotes w/o Lag. Best values are bold.

	ETTh2		Weather	
Variant	MSE	MAE	MSE	MAE
GRAIL	0.3118	0.3587	0.1909	0.2380
Base only	0.3092	**0.3553**	0.1914	0.2399
Uniform fusion	0.3107	0.3563	0.1943	0.2421
Shared fusion	0.3107	0.3564	0.1915	0.2398
Dense attention	0.3106	0.3566	0.1864	**0.2338**
Single lag	**0.3090**	0.3559	**0.1863**	0.2350

B.3 Sensitivity to Lag Count and Input Length

We vary $K \in \{3, 6, 9, 12\}$ with $L = 96$, and $L \in \{48, 96, 192\}$ with $K = 3$, on ETTh2 and Weather at $H = 96$. Each changed configuration is trained independently. The shared base checkpoint supplies the $L = 96$, $K = 3$ reference for both studies.

Table A7: Sensitivity results at $H = 96$. The base row is shared by both studies; all other rows change either K or L.

			ETTh2		Weather	
Configuration	L	K	MSE	MAE	MSE	MAE
Base	96	3	0.3118	0.3587	0.1909	0.2380
Lag	96	6	0.3102	0.3569	0.1935	0.2400
Lag	96	9	0.3108	0.3568	0.1772	0.2231
Lag	96	12	0.2998	0.3495	0.1965	0.2431
Context	48	3	0.3152	0.3530	0.1900	0.2336
Context	192	3	0.3258	0.3661	0.1834	0.2365

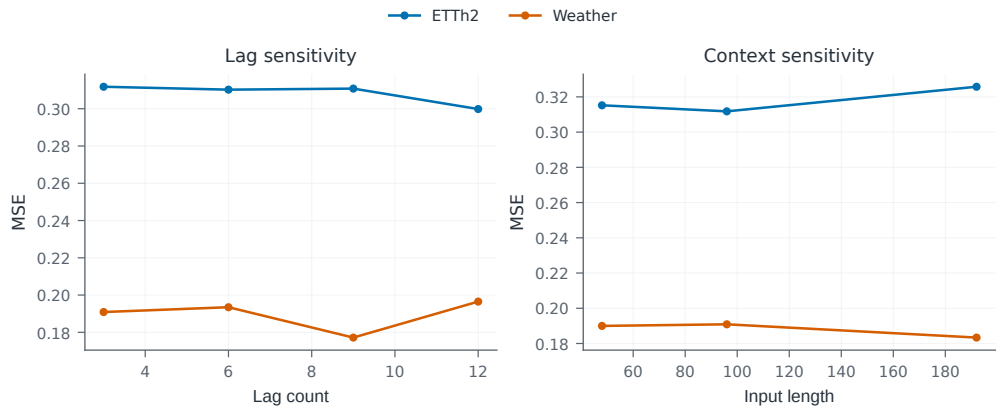


Figure A3: MSE as a function of lag count (left) and input length (right). The same base result is used in both panels.

Lag coverage. ETTh2 obtains its lowest tested MSE at $K = 12$ (0.2998), whereas Weather reaches its minimum at $K = 9$ (0.1772). Increasing Weather's lag count to 12 raises MSE to 0.1965. The default $K = 3$ supplies a common reference, rather than the best setting for every dataset. These additional runs extend the main-text sensitivity analysis: broader lag coverage can help, but its benefit is not monotonic.

Historical context. ETTh2 has its lowest MSE at $L = 96$, while Weather favors $L = 192$ (0.1834). MAE favors $L = 48$ on both datasets. These differences show that longer context does not uniformly improve forecasting, and the preferred input length depends on the evaluation metric as well as the dataset.

C Forecasting and Model Behavior

C.1 Matched Forecasting Examples

Figures A4 and A5 compare matched channels and origins across three variants. We display the first and last of 32 uniformly spaced test origins, using the channel with the largest training-set raw standard deviation. ETTh2 predictions are broadly similar; all three variants miss transient peaks in the Weather examples.

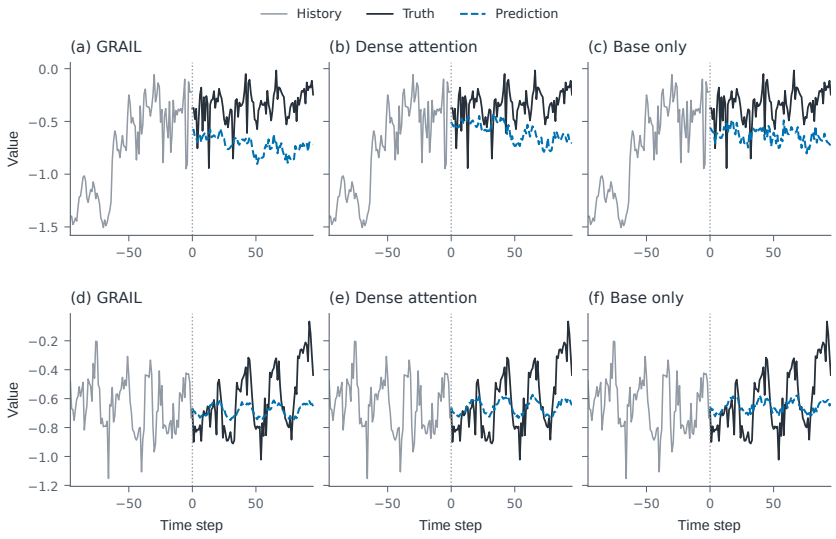


Figure A4: ETTh2 examples at $H = 96$. Gray shows history, black shows observations, and dashed blue shows predictions. The dotted line marks forecast step zero; values are training-standardized.

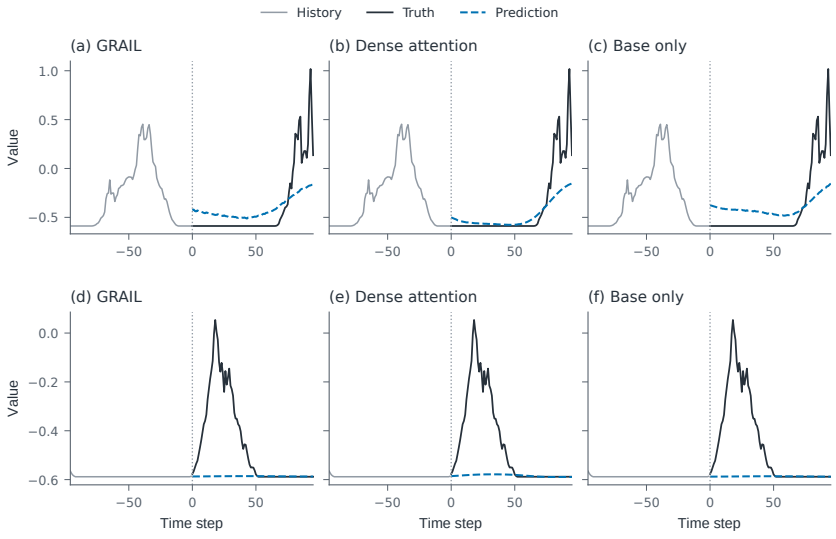


Figure A5: Weather examples with the same selection rule. Vertical scales are shared across models within each row. All three variants underestimate the transient peaks in these examples.

C.2 Error across Forecast Steps

Figure A6 averages squared error over the 32 sampled windows and up to 12 displayed channels at each position within an $H = 96$ forecast. This diagnostic subset differs from the full-test evaluation in Table A6.

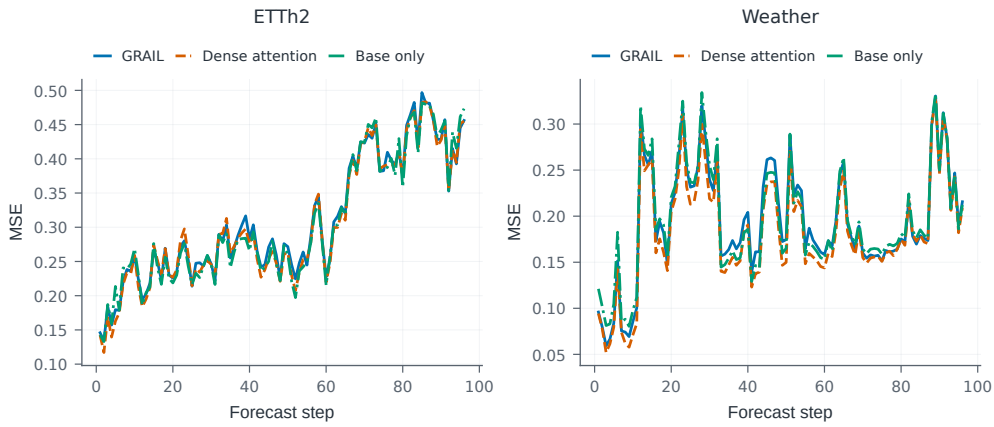


Figure A6: MSE at each forecast step on the sampled windows and channels. All variants use the same evaluation subset.

The curves overlap substantially. Error generally rises toward later steps on ETTh2, while Weather exhibits pronounced local peaks. The relative ordering changes across forecast positions, providing no uniform per-step advantage for a single variant in these examples.

C.3 Temporal Routing across Windows

Figure A7 shows routing weights averaged over all channels at each sampled origin. The long-scale branch receives the largest mean weight on both datasets, and Weather places less weight on the short-scale branch than ETTh2.

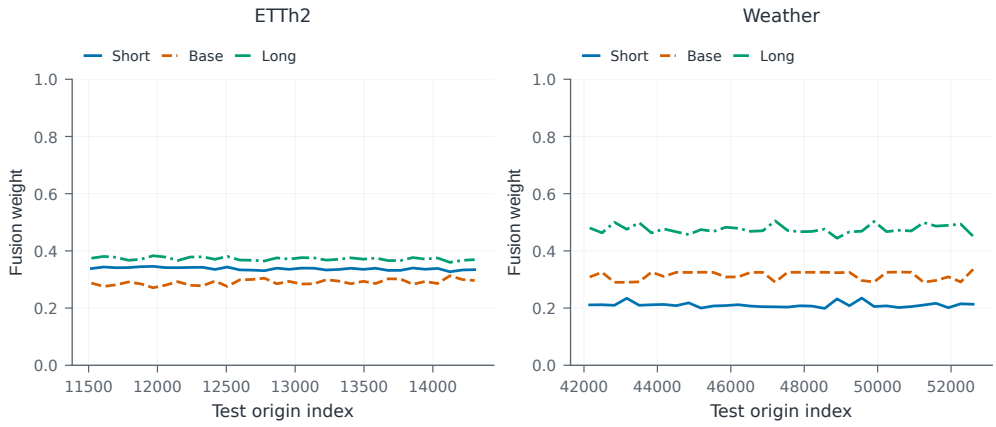


Figure A7: Channel-averaged TSF weights across sampled test origins. Lines connect the sampled windows in temporal order.

Weights vary across windows, but the branch ordering remains relatively stable. The router therefore exhibits input-dependent variation in these cases without frequent changes in its dominant temporal scale. Channel averaging may also conceal variation among individual variables.

C.4 Lagged Relationships and Channel Selection

Figure A8 examines the three displayed non-self sources with the largest mean attention to the example target. Pearson correlations pool historical observations across sampled windows; positive lags indicate that a source precedes the target. Source selection is restricted to the displayed channel subset.

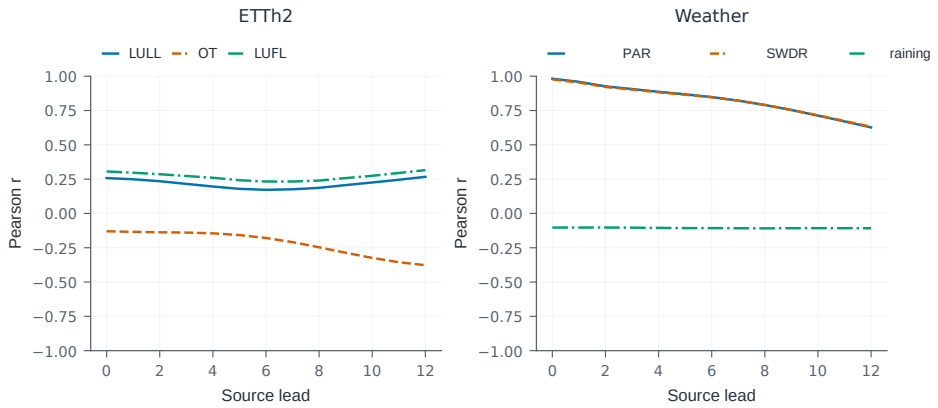


Figure A8: Historical correlation versus source lead, measured in dataset time steps. These descriptive associations do not identify causal effects.

PAR and SWDR remain positively correlated with the Weather target, whereas raining is close to zero. ETTh2 includes both positive and negative correlations. Selected sources therefore exhibit different historical relationships rather than uniformly high positive correlation.

The mask and attention differ because final aggregation also depends on the learned representations. The panels compare the two quantities from the same forward pass; attention is averaged across heads. Tick labels are omitted to emphasize the matrix structure.

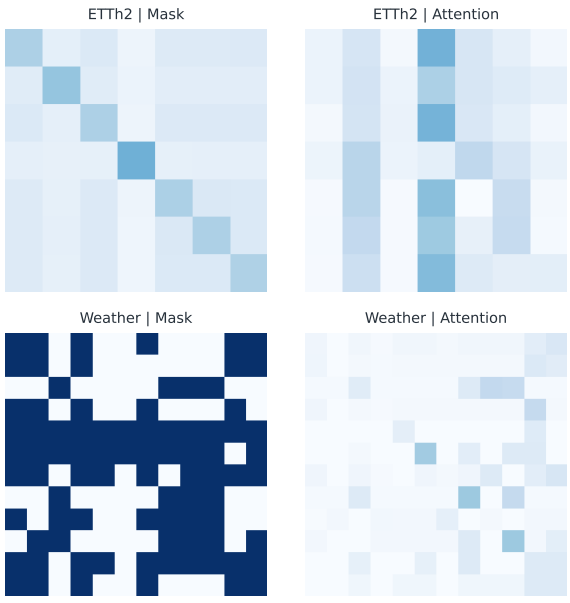


Figure A9: Predictive masks (left) and final-layer attention (right) at the first sampled origin. Top: ETTh2; bottom: Weather. Rows represent targets and columns sources in a matched order. Darker blue indicates larger values on a shared $[0, 1]$ scale.

D Computational Cost and Limitations

D.1 Measurement Protocol

We profile the trained GRAIL, dense-attention, and base-only models on ETTh2 and Weather at $H = 96$. All measurements use batch size 8 and float32 precision, with 10 warm-up steps followed by 50 timed steps. CUDA operations are synchronized for timing. A training step includes the forward pass, backpropagation, gradient clipping, and an Adam update. Memory is peak allocated GPU memory during that step.

Table A8: Computational cost per batch under the profiling protocol. Parameter counts are in millions, times in milliseconds, and memory in MiB.

Dataset	Variant	Params	Inference	Training	Memory
ETTh2	GRAIL	3.506	4.01	24.51	83.2
	Dense attention	3.504	2.91	22.43	83.2
	Base only	3.308	3.49	21.62	79.4
Weather	GRAIL	4.562	4.46	25.42	103.5
	Dense attention	4.554	2.69	21.91	103.3
	Base only	4.363	3.51	21.63	99.7

D.2 Runtime and Memory

GRAIL has a higher measured runtime than both controls. Relative to dense attention, inference time increases from 2.91 to 4.01 ms on ETTh2 and from 2.69 to 4.46 ms on Weather. Training-step time increases from 22.43 to 24.51 ms and from 21.91 to 25.42 ms, respectively. Parameter counts and peak allocated memory remain close to those of dense attention.

The main-text efficiency comparison concerns representative forecasting baselines; these measurements isolate the additional cost of TSF and GICM relative to simplified controls. Sparse mask entries do not imply execution with sparse kernels: the channel-attention computation remains dense and retains quadratic dependence on channel count. Timing results should therefore be interpreted for the measured implementation and batch configuration.

D.3 Scope and Limitations

The controlled experiments use one seed and two datasets, so they do not quantify initialization variability or establish component benefits across all benchmarks. The sensitivity results identify dataset-specific preferences among tested settings, rather than a universally optimal lag count or context length.

The visualizations are descriptive. Correlation curves and attention maps do not recover interventional causal effects, and the selected forecasting cases include missed transient peaks. The large-channel mask branch can be stochastic at inference, making a fixed evaluation seed important for repeatable exports.

Finally, the diagnostic error curves average a subset of windows and channels, while the reported forecasting metrics use the full test set. Their numerical values need not coincide. These distinctions delimit the conclusions supported by the supplementary experiments and preserve the separation between aggregate evaluation and qualitative model analysis.