WHEN CORRELATION IS NOT ENOUGH: UTILITY-DRIVEN VARIABLE ROUTING FOR TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate time-series forecasting commonly models variable dependencies through statistical correlations. However, correlation does not necessarily translate into predictive gains, and fixed relationships cannot readily adapt to changing system states. To address these limitations, we propose TimeSpur, a time-dependent sparse predictive-utility routing framework. TimeSpur first establishes a stable independent forecasting basis from each variable's own history and then constructs a sparse directed candidate graph using lagged relationships estimated from the training data. During training, paired stochastic masking measures the conditional predictive utility of a source variable for all targets against the observed future and uses this signal to calibrate a state-aware router. The model activates only candidate edges with positive utility under the current state and incorporates the resulting cross-variable information into the base prediction as a sparse residual. A single masking operation provides supervision from one source to all targets, avoiding exhaustive variable-wise evaluation in high-dimensional data. At inference, TimeSpur relies solely on the observed history, requiring neither masking nor future observations. Experiments on eight benchmark datasets demonstrate that TimeSpur achieves stable and competitive forecasting performance while controlling computational overhead.

Resources: https://anonymous.4open.science/r/TimeSpur-0FB4

1 INTRODUCTION

Multivariate time-series forecasting is fundamental to traffic management (He et al., 2025; Shao et al., 2026; Qiu et al., 2024), energy scheduling (Yi et al., 2024; Yu et al., 2024; Wu et al., 2026b), and environmental monitoring (Ye et al., 2025; Zhou et al., 2026). Unlike univariate forecasting, multivariate data contain not only the intrinsic trends and periodic patterns of individual variables, but also potential information transfer across variables (Zhang et al., 2026a; Chen et al., 2024b). For example, changes in upstream traffic flow may precede their effects on downstream roads, while the consumption patterns of certain users may reflect collective variations in regional electricity demand (Zhang et al., 2026b). Consequently, recent studies have increasingly focused on modeling cross-variable dependencies and have achieved promising forecasting performance.

However, these methods commonly rely on an assumption that deserves closer examination: if two variables are statistically correlated, propagating information between them should benefit future prediction (Yang et al., 2024; Kim & Jo, 2026). Although intuitive, this assumption overlooks several fundamental complexities of cross-variable modeling. It remains unclear whether statistical correlation corresponds to genuine predictive value, whether variable dependencies change with the system state, and whether an individual propagation edge actually improves forecasting (Qiu et al., 2026). Therefore, the central question is not merely *which variables are correlated*, but rather *which variables are truly useful for prediction under the current state*. We examine this issue from the following three perspectives.

1) Discrepancy between statistical correlation and predictive value. Existing graph-based methods commonly select variable relations according to correlation coefficients, representation similarity, or attention weights. As shown in Fig. 1(a), highly correlated variables may merely share

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

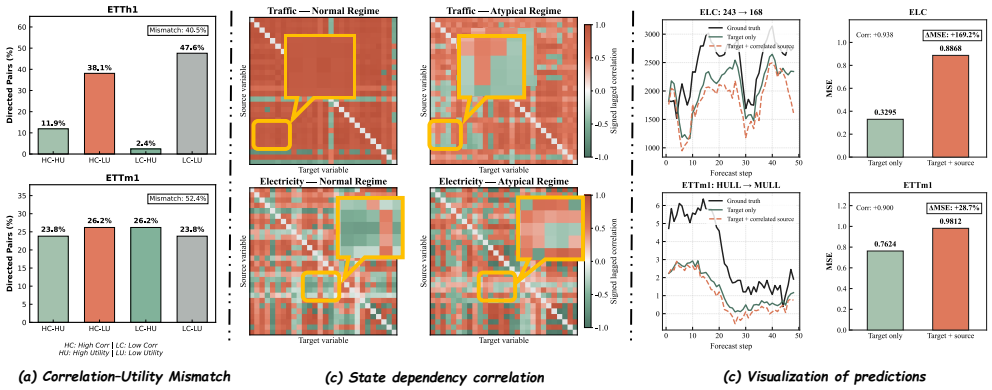


Figure 1: **Correlation does not imply predictive utility.** (a) Correlation–utility mismatch on ETT datasets. (b) State-dependent variable relations in Traffic and Electricity. (c) Highly correlated sources can increase forecasting errors.

similar periodic patterns without providing additional predictive information. Conversely, variables with weak overall correlation may offer informative leading signals through directional and lagged relations. Thus, useful information often originates from a small number of variables that change in advance, rather than from variables that move synchronously with the prediction target.

2) Fixed variable graphs cannot capture state-dependent relations. Many methods employ the same adjacency structure for all samples or require each target variable to aggregate a fixed number of neighbors. In real-world systems, however, cross-variable relations vary with the operating state. As illustrated in Fig. 1(b), Traffic exhibits strong and relatively consistent positive lagged correlations among many roads during normal periods. Under atypical conditions, these relations weaken substantially, some of them reverse direction, and the overall structure becomes more localized. Similarly, Electricity shows relatively stable collective dependencies during normal periods, whereas its dependency structure is markedly reorganized under abnormal or rapidly changing loads. These observations suggest that whether an edge should participate in information propagation depends not only on the identities of its source and target, but also on the current historical state. A fixed graph may therefore preserve outdated relations while overlooking newly emerging informative dependencies.

3) Lack of direct supervision for edge-level predictive gains. The overall forecasting loss evaluates only the final prediction and cannot determine whether a particular cross-variable edge provides useful information or introduces redundancy and noise. Likewise, a large attention or graph weight merely indicates that the model tends to use a variable; it does not demonstrate that incorporating this variable improves the target forecast. As shown in Fig. 1(c), although the selected source variables are highly correlated with their targets, adding them to the target histories increases the forecasting error. Without outcome-based supervision for individual edges, a model may therefore retain relations that appear statistically relevant but are unhelpful or even detrimental to prediction.

To address these limitations, we propose **TimeSpur**, a predictive-utility-driven dynamic sparse variable routing framework. The model first captures the intrinsic evolution of each variable to establish a stable independent forecasting basis, ensuring reliable predictions without relying on cross-variable information. It then constructs a sparse set of candidate edges for each target from lagged relations in the training data. Correlation is used only to restrict the search space rather than as the final criterion for determining whether an edge is useful. To directly measure the predictive contribution of a candidate source, we randomly mask one source variable during training and perform complete-input and masked-input forecasts over the same historical window. By evaluating both forecasts against the observed future, the model attributes the prediction deviation caused by the missing source and identifies the incremental information that cannot be replaced by the remaining input.

The utility signals obtained through paired masking are further used to supervise a state-conditioned utility router, which estimates the predictive value of each candidate edge for the current sample.

Only candidate edges expected to provide positive predictive gains are activated, while uninformative or potentially harmful relations are suppressed. The resulting propagation structure is therefore dynamically sparse rather than constrained to a fixed number of active neighbors. During stable periods, the model can rely primarily on each target's own history; during load fluctuations or emerging traffic congestion, it can activate a small number of informative directional relations. At inference time, the router selects edges solely from the observed history without accessing future observations, thereby avoiding future information leakage. More importantly, the proposed method explicitly distinguishes whether two variables are statistically related from whether information should actually be propagated between them.

Our main contributions are summarized as follows:

- ❶ **New perspective.** We characterize cross-variable dependencies through conditional predictive utility rather than treating statistical association as direct evidence of information contribution. This perspective distinguishes superficial correlation from predictive information that is complementary to the target's own history.
- ❷ **Technical contribution.** We propose a predictive-utility-driven dynamic sparse routing method. Paired stochastic masking extracts edge-level conditional utility supervision from observed futures, which calibrates a state-conditioned router to activate only candidate directional edges with positive predictive contributions.
- ❸ **Comprehensive evaluation.** Experiments on multiple multivariate forecasting benchmarks demonstrate stable and competitive forecasting performance with modest computational overhead. The learned variable relations can also be validated through the error differences induced by source masking.

2 Related Work

Cross-variable Modeling for Time-Series Forecasting. A central question in multivariate time-series forecasting is whether and how information should be exchanged across variables. Channel-independent methods model each series separately, preserving variable-specific dynamics and avoiding interference from unrelated channels. However, the absence of cross-variable communication may discard complementary signals. Channel-dependent methods instead capture inter-series interactions through attention, mixing, or graph propagation. Representative examples include iTransformer (Liu et al., 2024), MSGNet (Cai et al., 2024), TimeMosaic (Ding et al., 2025), SOFTS (Han et al., 2024),Hi-patch (Luo et al., 2025), ExoTimer (Wu et al., 2026a),CompRestacking (Kim & Jo, 2026) and HyperMixer (Tian et al., 2025). Other studies further explore leading indicators (Zhao & Shen, 2024) and variable-dependent temporal delays (Ma et al., 2025). Although these methods enrich multivariate representations, extensively mixing channels may also propagate redundant or spurious information, particularly when variable dependencies change across samples.

Selective Dependency Modeling. Recent studies have therefore shifted from unrestricted interaction toward selective dependency modeling. Channel clustering groups variables according to their similarity (Chen et al., 2024a), while sparse interaction networks (Zhang et al., 2025) and variate-reduction strategies (Kang et al., 2025) restrict unnecessary communication. TimeFilter (Hu et al., 2025) further performs patch-specific graph filtration to retain fine-grained spatial–temporal dependencies. These approaches reduce indiscriminate channel mixing, but their selection criteria are primarily derived from similarity, representation compatibility, or learned graph scores. Such quantities characterize whether two variables are related, but do not directly verify whether transmitting information along a particular edge improves the target forecast.

Different from these methods, TimeSpur separates dependency discovery from dependency validation. Lagged correlation is used only to construct a sparse candidate set, whereas paired stochastic masking provides outcome-grounded supervision for measuring the conditional predictive utility of each source–target relation. A state-aware router then activates only positive-utility candidate edges under the current historical context. In this way, TimeSpur models cross-variable dependencies according to their realized predictive value rather than statistical association alone.

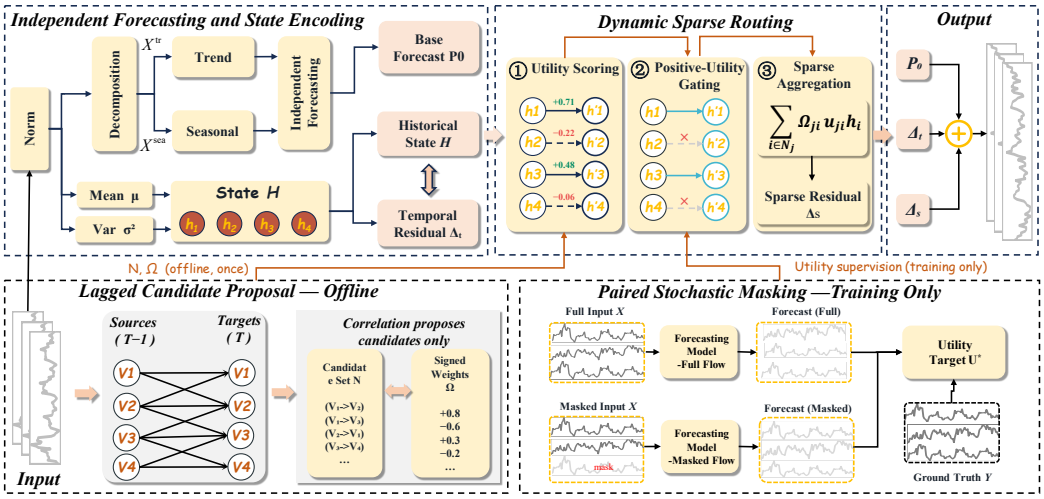


Figure 2: Overall architecture of TimeSpur. (i) **Independent Forecasting** models each variable separately to produce the base forecast and historical state. (ii) **Lagged Candidate Proposal** constructs a sparse directed candidate graph from the training data offline. (iii) **Paired Stochastic Masking** derives edge-level utility supervision from full and masked forecasts during training. (iv) **Dynamic Sparse Routing** activates positive-utility edges and aggregates their information as a residual correction using only observed history at inference.

3 METHOD

3.1 STRUCTURAL OVERVIEW

As illustrated in Fig. 2, our proposed TimeSpur consists of four components: (i) **Independent Forecasting Branch** models the intrinsic evolution of each variable through temporal decomposition and variable-specific correction, producing a stable base forecast and state representation. (ii) **Lagged Candidate Proposal** constructs a sparse directed candidate graph once from the training split, where lagged correlation only restricts the potential propagation space. (iii) **Paired Stochastic Masking** measures the conditional predictive utility of a sampled source variable for all targets and provides direct supervision for utility estimation during training. (iv) **Dynamic Sparse Router** predicts candidate-edge utility from the current historical state, activates only positive-utility edges, and incorporates the aggregated cross-variable information as a residual correction. During inference, TimeSpur performs routing using only the observed history, without masking or future observations.

3.2 PROBLEM FORMULATION

Given a historical multivariate sequence $X \in \mathbb{R}^{B\times L\times C}$ and the available future time covariates T^f, multivariate time-series forecasting is formulated as

$$\widehat{Y} = \mathcal{F}_\theta(X, T^f) \in \mathbb{R}^{B\times H\times C}. \tag{1}$$

Here, B, L, H, and C denote the batch size, look-back length, forecast horizon, and number of variables, respectively. $\mathcal{F}_\theta$ is the forecasting model parameterized by θ, and $\widehat{Y}$ predicts the future observations $Y \in \mathbb{R}^{B\times H\times C}$.

3.3 INDEPENDENT FORECASTING AND STATE ENCODING

We first apply instance normalization to each variable along the temporal dimension: $\widetilde{X} = (X - \mu)/\sigma$, where $\mu, \sigma \in \mathbb{R}^{B\times 1\times C}$ are the historical mean and standard deviation, and $\widetilde{X} \in \mathbb{R}^{B\times L\times C}$ is the normalized input. These statistics are retained for state encoding and final denormalization.

In the normalized space, shared temporal mappings capture common dynamics, while a low-rank variable modulation preserves variable-specific evolution:

$$\begin{aligned} X^{\mathrm{tr}} &= \mathrm{MA}(\widetilde{X}), \qquad X^{\mathrm{sea}} = \widetilde{X} - X^{\mathrm{tr}}, \\ P_0 &= \Big[f_{\mathrm{tr}}\big((X^{\mathrm{tr}})^\top\big) + f_{\mathrm{sea}}\big((X^{\mathrm{sea}})^\top\big) + \big((\widetilde{X}^\top W_1) \odot \Gamma\big) W_2 \Big] \odot (1 + A) + B. \end{aligned} \tag{2}$$

Here, $\mathrm{MA}(\cdot)$ denotes moving-average decomposition, and $f_{\mathrm{tr}}, f_{\mathrm{sea}} : \mathbb{R}^L \to \mathbb{R}^H$ forecast the smooth and fluctuating components, respectively. $W_1 \in \mathbb{R}^{L \times R_i}$ and $W_2 \in \mathbb{R}^{R_i \times H}$ form the low-rank temporal mapping, while $\mathbf{\Gamma} \in \mathbb{R}^{C \times R_i}$ provides variable-specific modulation. $A, B \in \mathbb{R}^{C \times H}$ calibrate the gain and bias of each variable at each forecast step. Consequently, $P_0 \in \mathbb{R}^{B \times C \times H}$ provides an independent forecast without cross-variable propagation.

To condition variable interactions on the current operating regime, we jointly encode the normalized temporal pattern, level, and variability:

$$H = \mathrm{LN}\Big(\mathrm{GELU}\Big[\widetilde{X}^\top W_h + [\boldsymbol{\mu}, \log \sigma] W_l\Big]\Big) \in \mathbb{R}^{B \times C \times D}, \qquad \Delta_t = H W_p \in \mathbb{R}^{B \times C \times H}. \tag{3}$$

The projection matrices satisfy $W_h \in \mathbb{R}^{L \times D}$, $W_l \in \mathbb{R}^{2 \times D}$, and $W_p \in \mathbb{R}^{D \times H}$, where D is the state dimension. H represents the current state of each variable, and Δ_t is its state-dependent temporal correction. This branch establishes a stable variable-wise forecast, but does not yet determine which external variables provide non-redundant predictive information.

3.4 Conditional Predictive Utility Learning

3.4.1 Candidate Graph Construction

To preserve potential directional relations while controlling the search cost, we construct a candidate graph using only the training split. Let $Z \in \mathbb{R}^{T \times C}$ denote the training sequence. We standardize the two adjacent temporal slices $Z_{1:T-1}$ and $Z_{2:T}$ column-wise, obtaining $\overline{Z}^-$ and $\overline{Z}^+$. The one-step lagged relation matrix and its candidate graph are defined as

$$\begin{aligned} R &= \frac{(\overline{Z}^+)^\top \overline{Z}^-}{T-1} \in \mathbb{R}^{C \times C}, \\ N &= \mathrm{TopK}_{i \neq j}(|R|, K) \in \mathbb{N}^{C \times K}, \\ \Omega &= \mathrm{RowNorm}\big(\mathrm{Gather}(R, N)\big) \in \mathbb{R}^{C \times K}. \end{aligned} \tag{4}$$

Rows and columns of R correspond to target and source variables, respectively; hence, each off-diagonal entry represents a temporally directed relation from a source at the preceding step to a target at the subsequent step. After excluding self-connections, N stores the indices of the K candidate sources for each target, and Ω contains their signed, row-normalized weights. The graph is constructed once before training and only restricts the search space; membership in the candidate set does not imply that an edge should always be active.

3.4.2 Utility Estimation via Paired Stochastic Masking

For each training instance, the model uniformly samples a source variable s from the C variables and removes both its normalized history and local statistics, yielding the masked input X^{-s}. Let $\widehat{Y}$ and $\widehat{Y}^{-s}$ denote the forecasts obtained from the complete and masked inputs, respectively. The conditional predictive utility of source variable s is defined as

$$u_s^* = \tanh\left(\frac{\mathcal{E}_H\Big(\widehat{Y}^{-s}, Y\Big) - \mathcal{E}_H\Big(\widehat{Y}, Y\Big)}{\tau}\right) \in \mathbb{R}^C, \tag{5}$$

where $\mathcal{E}_H(\cdot, \cdot)$ denotes the target-wise mean squared error averaged over the forecasting horizon, and τ is a scale factor updated using a moving average of the complete-input forecasting error. A positive component of u_s^* for target variable j indicates that masking source variable s increases the forecasting error of j, suggesting that s provides additional predictive information for this target. Each masking operation simultaneously measures the utility of s for all target variables. During

mini-batch training, a source variable is independently sampled for each instance; therefore, a single additional masked forward pass produces the utility supervision $U^* \in \mathbb{R}^{B\times C}$ without exhaustively evaluating all C source variables.

Since the utility labels derived from paired masking are evaluated against the observed future, they are used exclusively to calibrate the router during training. To translate this outcome-level supervision into an executable edge-selection mechanism at inference time, the model further learns a state-conditioned utility function that estimates the predictive value of each candidate edge solely from the observed history.

3.5 Utility-Driven Dynamic Sparse Routing

The utility router maps each variable state into two asymmetric spaces, representing its roles as an information source and a prediction target:

$$Q_s = \tanh(HW_s), \qquad Q_t = \tanh(HW_t), \qquad W_s, W_t \in \mathbb{R}^{D\times R_u}. \tag{6}$$

$Q_s, Q_t \in \mathbb{R}^{B\times C\times R_u}$ are the source and target representations, and R_u is the utility embedding dimension. All edge quantities below are instance-dependent; we omit the batch index for clarity. For each candidate edge $i \in \mathcal{N}_j$, the predicted utility and its dynamic gate are

$$\widehat{u}_{ji} = \frac{q_{s,i}^\top q_{t,j}}{\sqrt{R_u}}, \qquad \lambda_{ji}^h = \mathbb{I}(\widehat{u}_{ji} > 0), \qquad \lambda_{ji} = \lambda_{ji}^h + \sigma(\widehat{u}_{ji}) - \mathrm{sg}[\sigma(\widehat{u}_{ji})]\,. \tag{7}$$

Here, $\widehat{u}_{ji}$ is the state-conditioned utility from source i to target j. The scores of all candidate edges form $\widehat{U}_{\mathcal{N}} \in \mathbb{R}^{B\times C\times K}$. $\mathbb{I}(\cdot)$, $\sigma(\cdot)$, and $\mathrm{sg}[\cdot]$ denote the indicator, sigmoid, and stop-gradient operators, respectively. In the forward pass, λ_{ji} is exactly the binary gate λ_{ji}^h; in the backward pass, the continuous term provides gradients to the router. Consequently, the effective number of neighbors is determined by the positive-utility edges in each instance rather than by a fixed routing budget.

The activated source states are aggregated using their signed directional weights and incorporated as a residual correction:

$$S_j = \sum_{i\in\mathcal{N}_j} \omega_{ji}\lambda_{ji}H_i, \qquad \Delta_s = f_s(\tanh S) \in \mathbb{R}^{B\times C\times H}, \tag{8}$$

where ω_{ji} is drawn from Ω, $S \in \mathbb{R}^{B\times C\times D}$ is the dynamically aggregated state, and $f_s : \mathbb{R}^D \to \mathbb{R}^H$ projects it to the cross-variable forecasting residual. The final prediction is

$$\widetilde{Y} = (P_0 + \Delta_t + \Delta_s)^\top + P_{\mathrm{cal}}, \qquad \widehat{Y} = \widetilde{Y} \odot \sigma + \mu. \tag{9}$$

$P_{\mathrm{cal}} \in \mathbb{R}^{B\times H\times C}$ is an optional calendar residual derived from known future time covariates and is omitted when valid time covariates are unavailable. $\widetilde{Y}$ and $\widehat{Y}$ denote the forecasts in the normalized and original scales, respectively.

3.6 Optimization Objective

We jointly optimize forecasting accuracy and utility estimation so that the routing decisions are constrained by both the final forecasting outcome and the measured edge-level utility:

$$\begin{aligned} \mathcal{L}_{\mathrm{pred}} &= \frac{1}{BHC}\left\|\widehat{Y} - Y\right\|_F^2, \\ \mathcal{L}_{\mathrm{util}} &= \frac{1}{BC}\left\|\widehat{U}_s - \mathrm{sg}[U_s^*]\right\|_F^2, \\ \mathcal{L} &= \mathcal{L}_{\mathrm{pred}} + \mathcal{L}_{\mathrm{util}}. \end{aligned} \tag{10}$$

Here, $\widehat{U}_s, U_s^* \in \mathbb{R}^{B\times C}$ denote the predicted and measured utility rows for the source sampled in each instance. $\mathcal{L}_{\mathrm{pred}}$ optimizes the complete-input forecast, whereas $\mathcal{L}_{\mathrm{util}}$ calibrates the mapping from historical states to predictive utility.

Utility-target construction is detached from the computational graph, and the calibration branch receives stop-gradient historical states. Therefore, $\mathcal{L}_{\mathrm{util}}$ calibrates the utility scorer without directly perturbing the independent historical representation. Paired masking is used only to generate supervision during training. At inference, the router estimates candidate-edge utility directly from observed history, requiring neither additional masked forecasts nor future observations.

Models	TimeSpur (Ours)		XLinear (AAAI'2026)		EMAformer (AAAI'2026)		TimeFilter (ICML'2025)		PatchMLP (AAAI'2025)		iTransformer (ICLR'2024)		TimeMixer (ICLR'2024)		PatchTST (ICLR'2023)		DLinear (AAAI'2023)		FreTS (NIPS'2023)		LightTS (2022)	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.419**	**0.431**	0.424	0.435	0.459	0.455	0.511	0.482	0.462	0.460	0.487	0.474	0.430	0.438	0.503	0.485	0.429	0.444	0.461	0.465	0.503	0.496
ETTh2	**0.342**	**0.386**	0.349	0.391	0.395	0.416	0.402	0.421	0.380	0.415	0.411	0.424	0.358	0.396	0.398	0.426	0.492	0.478	0.495	0.483	0.602	0.519
ETTm1	0.353	0.379	**0.344**	**0.377**	0.369	0.392	0.399	0.408	0.374	0.396	0.386	0.406	0.356	0.385	0.372	0.399	0.357	0.380	0.369	0.391	0.388	0.412
ETTm2	**0.258**	**0.310**	0.259	0.317	0.272	0.324	0.278	0.330	0.269	0.324	0.292	0.338	0.265	0.323	0.277	0.332	0.282	0.348	0.284	0.339	0.347	0.392
Electricity	**0.156**	**0.255**	0.164	0.259	0.162	0.255	0.167	0.260	0.200	0.301	0.180	0.276	0.173	0.267	0.173	0.272	0.170	0.269	0.171	0.270	0.207	0.316
Weather	**0.222**	0.269	0.225	**0.264**	0.229	0.267	0.231	0.270	0.230	0.267	0.237	0.272	0.227	0.266	0.236	0.271	0.245	0.299	0.230	0.276	0.230	0.281
Solar-Energy	**0.192**	**0.249**	–	–	0.204	0.260	0.200	0.255	0.250	0.292	0.232	0.280	0.218	0.276	0.211	0.262	0.255	0.315	0.212	0.267	0.217	0.287
Traffic	0.437	0.297	0.422	0.292	0.420	0.296	**0.420**	**0.284**	0.524	0.382	0.463	0.335	0.444	0.316	0.428	0.297	0.445	0.308	0.461	0.313	0.507	0.369

Table 1: Long-term forecasting results. The input length L is 320. All results are averaged across four prediction lengths $H \in \{96, 192, 336, 720\}$. The best result is **red**, and the second-best result is blue. See Table 4 for the full results.

4 EXPERIMENT

Datasets. We evaluate our method on eight widely used multivariate time-series forecasting benchmarks: ETTh1, ETTh2, ETTm1, ETTm2, Electricity, Weather, Solar-Energy, and Traffic. These datasets cover diverse application domains, temporal resolutions, and numbers of variables, including electricity consumption, meteorological observations, photovoltaic power generation, and road traffic conditions. Further details are provided in Appendix B.

Baselines. We compare the proposed method with ten representative forecasting models, including XLinear (Chen et al., 2026), EMAformer (Zhang et al., 2026c), TimeFilter (Hu et al., 2025), PatchMLP (Tang & Zhang, 2025), iTransformer (Liu et al., 2024), TimeMixer (Wang et al., 2024), PatchTST (Nie et al., 2023), DLinear (Zeng et al., 2023), FreTS (Yi et al., 2023), and LightTS (Zhang et al., 2022).

Implementation Details. All experiments are implemented in PyTorch Paszke (2019) and conducted on a single NVIDIA RTX 5080 GPU with 16 GB of memory. Following the standard evaluation protocol for time-series forecasting, we adopt mean squared error (MSE) and mean absolute error (MAE) as the evaluation metrics, where lower values indicate better forecasting performance. Further details on the hyperparameter settings are provided in Appendix D.

4.1 MULTIVARIATE FORECASTING

Table 1 presents the forecasting results averaged over four prediction horizons, $H \in \{96, 192, 336, 720\}$, with the look-back length fixed to $L = 320$. The best and second-best results are highlighted in **red** and blue, respectively.

Overall Performance. TimeSpur achieves the best performance in 11 out of 16 dataset–metric comparisons. It ranks first in both MSE and MAE on ETTh1, ETTh2, ETTm2, Electricity, and Solar-Energy, and obtains the best MSE on Weather. These results demonstrate its consistent forecasting performance across datasets with diverse sampling frequencies, temporal characteristics, and numbers of variables.

Performance on High-Dimensional Data. The advantages of TimeSpur are particularly evident on datasets with substantial cross-variable interactions. Compared with the strongest competing methods, TimeSpur reduces the average MSE by approximately 3.7% and 4.3% on Electricity and Solar-Energy, respectively. These improvements support the effectiveness of separating correlation-based candidate construction from predictive-utility-based edge activation, thereby preventing statistically correlated but predictively uninformative variables from being indiscriminately propagated.

Performance in Challenging Settings. TimeSpur remains competitive when its advantages are less pronounced. On ETTm1, it ranks second in both MSE and MAE, with only a small gap from the strongest baseline. TimeFilter achieves better results on Traffic, possibly because this dataset contains 862 sensor variables with strong periodic synchronization and localized interactions, which can be effectively captured by its fine-grained dependency filtering mechanism.

Dataset	Metric	Routing Design			Candidate Proposal		Full Model
		Independent	Static Correlation	Dynamic w/o Utility	Random Candidates	Undirected Candidates	**TimeSpur (Full)**
ETTm1	MSE ↓	0.297	0.294	0.298	0.296	0.295	**0.291**
	MAE ↓	0.247	0.244	0.248	0.248	0.245	**0.243**
Solar	MSE ↓	0.189	0.185	0.188	0.185	0.182	**0.177**
	MAE ↓	0.247	0.241	0.246	0.244	0.243	**0.239**
Weather	MSE ↓	0.150	0.149	0.153	0.155	0.151	**0.146**
	MAE ↓	0.208	0.207	0.209	0.210	0.213	**0.199**
Electricity	MSE ↓	0.141	0.145	0.137	0.139	0.140	**0.130**
	MAE ↓	0.235	0.233	0.232	0.235	0.231	**0.225**

Table 2: Ablation results of TimeSpur on four representative datasets. Routing and candidate-graph variants are grouped according to the modified component.

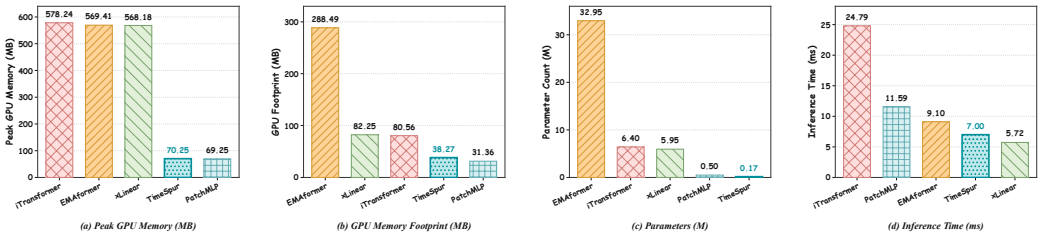


Figure 3: **Efficiency comparison on Electricity.** We compare (a) peak GPU memory, (b) GPU memory footprint, (c) parameter count, and (d) inference latency. All models are profiled with a batch size of 32 on a single NVIDIA RTX 5080 GPU with 16 GB memory. Lower values indicate better efficiency.

4.2 Ablation Study

We compare TimeSpur with five ablation variants. **Independent** removes the candidate graph and cross-variable propagation. **Static-Corr** retains the directed lagged graph but keeps all candidate edges active without dynamic utility gating. **Dynamic w/o Utility** preserves state-dependent routing but removes paired stochastic masking, allowing the router to learn only from the forecasting loss. **Random Candidates** replaces lagged-correlation candidates with an equally sparse random graph while retaining utility supervision. **Undirected Candidates** uses synchronous correlation instead of directed lagged relations, with the remaining routing mechanism unchanged.

As shown in Table 2, the complete TimeSpur performs best across all datasets. The competitive **Independent** results demonstrate the strength of the forecasting backbone, while the full model confirms the benefit of controlled cross-variable propagation. **Static-Corr** shows that correlation provides useful candidates, but fixed propagation cannot adapt to changing states. **Dynamic w/o Utility** verifies the importance of direct masking-based supervision. The degradation under **random** and **undirected candidates** further demonstrates that both meaningful candidate selection and temporal direction are necessary. These components therefore provide complementary contributions to TimeSpur.

4.3 Efficiency Analysis

Experimental setup. We evaluate computational efficiency on Electricity under an identical hardware and inference configuration. For a fair comparison, the batch size is fixed to 32 for all methods. As shown in Figure 3, TimeFilter cannot complete inference on the 16 GB GPU because of an out-of-memory error and is therefore omitted.

Efficiency comparison. TimeSpur contains only 0.17M parameters, the fewest among all compared methods and 66.4% fewer than PatchMLP, the second-smallest model. Its inference latency is 7.00 ms, ranking second only to xLinear while remaining faster than EMAformer, PatchMLP, and iTransformer. TimeSpur also requires only 38.27 MB of GPU memory, with a peak usage of 70.25 MB, both of which are close to the lowest values achieved by PatchMLP. These results demon-

strate that the state-dependent utility router and sparse cross-variable propagation introduce limited overhead, allowing TimeSpur to remain compact, fast, and memory efficient.

4.4 Case Study of Dynamic Sparse Routing

Figure 4 presents three representative Electricity samples to illustrate the effect of dynamic sparse routing. The pre-routing forecast retains the independent forecasting, temporal-state, and calendar components but excludes cross-variable propagation. The lower panels visualize the sparse residual introduced by TimeSpur. Although the pre-routing branch captures the main periodic patterns, it tends to underestimate several high-load intervals. TimeSpur generates horizon-dependent residuals to compensate for these deviations, bringing the final predictions closer to the ground truth. Accordingly, the normalized MSE decreases from 0.161 to 0.027, from 0.107 to 0.032, and from 0.195 to 0.066. These cases show that TimeSpur preserves a stable independent prediction while selectively incorporating useful cross-variable information as an adaptive residual.

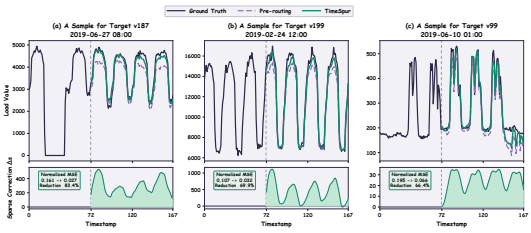


Figure 4: **Dynamic sparse routing cases on Electricity.** TimeSpur improves the pre-routing forecasts by introducing state-dependent sparse residuals.

4.5 State-Dependent Routing Dynamics

To examine how TimeSpur adapts variable dependencies to changing states, Figure 5 visualizes the predicted utility scores of representative candidate edges on Electricity. For each target, the upper curve shows its evolving historical state, while the heatmap presents the utility of different candidate sources over consecutive test windows. Positive scores activate the corresponding edges, whereas non-positive scores suppress information propagation. The results reveal clear temporal and target-specific variations: the same candidate edge may alternate between active and inactive states, while different targets exhibit distinct routing patterns over the same period. Some sources provide persistent information, whereas others are useful only in particular regimes. This confirms that the offline candidate graph only defines the possible propagation space, while the actual information flow is dynamically determined by the current history.

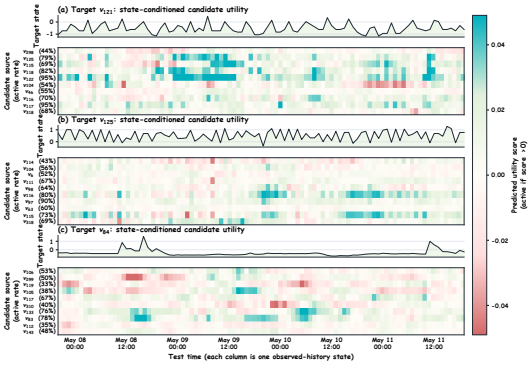


Figure 5: **State-dependent routing dynamics on Electricity.** Utility scores vary with the observed state, dynamically activating or suppressing candidate edges.

5 Conclusion

This paper introduces TimeSpur, a state-dependent sparse utility routing framework for multivariate time-series forecasting. By combining independent forecasting, lagged candidate graphs, paired stochastic masking, and dynamic edge activation, TimeSpur separates correlation-based candidate construction from predictive-utility estimation and propagates only cross-variable information that is beneficial under the current state. Experiments on eight benchmarks demonstrate its effectiveness. We believe this work offers a new perspective for reconsidering cross-variable dependency modeling through predictive gains rather than statistical correlations.

6 AI USE STATEMENT

In this work, generative AI tools were used only to assist with language polishing and improve the clarity and readability of the manuscript.

7 ETHICS STATEMENT

This work focuses on general time-series forecasting using public benchmarks and does not involve personal data or human subjects. We identify no direct ethical concerns.

8 REPRODUCIBILITY STATEMENT

The model architecture and objectives are formally defined in the main text. Dataset details, evaluation metrics, experimental settings, and hyperparameters are provided in the paper and appendix.

REFERENCES

Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pp. 11141–11149. AAAI Press, 2024. doi: 10.1609/AAAI.V38I10.28991. URL https://doi.org/10.1609/aaai.v38i10.28991.

Jialin Chen, Jan Eric Lenssen, Aosong Feng, Weihua Hu, Matthias Fey, Leandros Tassiulas, Jure Leskovec, and Rex Ying. From similarity to superiority: Channel clustering for time series forecasting. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 130635–130663. Curran Associates, Inc., 2024a. doi: 10.52202/079017-4152. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/eb9b18ccb76a1156af5779ffdca1d91f-Paper-Conference.pdf.

Xiaodan Chen, Xiucheng Li, Xinyang Chen, and Zhijun Li. Structured matrix basis for multivariate time series forecasting with interpretable dynamics. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 24326–24349. Curran Associates, Inc., 2024b. doi: 10.52202/079017-0766. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2b47305e1c81890b1089a405686ad183-Paper-Conference.pdf.

Xinyang Chen, Huidong Jin, Yu Huang, and Zaiwen Feng. Xlinear: a lightweight and accurate mlp-based model for long-term time series forecasting with exogenous inputs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20325–20335, 2026.

Kuiye Ding, Fanda Fan, Chunyi Hou, Zheya Wang, Lei Wang, Zhengxin Yang, and Jianfeng Zhan. Timemosaic: Temporal heterogeneity guided time series forecasting via adaptive granularity patch and segment-wise decoding. *arXiv preprint arXiv:2509.19406*, 2025.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. Softs: Efficient multivariate time series forecasting with series-core fusion. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 64145–64175. Curran Associates, Inc., 2024. doi: 10.52202/079017-2046. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/754612bde73a8b65ad8743f1f6d8ddf6-Paper-Conference.pdf.

Sicheng He, Junzhong Ji, and Minglong Lei. Decomposed spatio-temporal mamba for long-term traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 11772–11780, 2025.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. Timefilter: Patch-specific spatial-temporal graph filtration for time series forecasting. *arXiv preprint arXiv:2501.13041*, 2025.

Junhyeok Kang, Yooju Shin, and Jae-Gil Lee. Vardrop: Enhancing training efficiency by reducing variate redundancy in periodic time series forecasting. In Toby Walsh, Julie Shah, and Zico Kolter (eds.), *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pp. 17742–17749. AAAI Press, 2025. doi: 10.1609/AAAI.V39I17.33951. URL https://doi.org/10.1609/aaai.v39i17.33951.

Min Kim and Ohyun Jo. Comprestacking: Capturing channel dependency in highly correlated multivariate time series data (student abstract). In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 41239–41241. AAAI Press, 2026. doi: 10.1609/AAAI.V40I48.42227. URL https://doi.org/10.1609/aaai.v40i48.42227.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Yicheng Luo, Bowen Zhang, Zhen Liu, and Qianli Ma. Hi-patch: Hierarchical patch gnn for irregular multivariate time series. In *Forty-second International Conference on Machine Learning*, 2025.

Xiaowen Ma, Zhen-Liang Ni, Shuai Xiao, and Xinghao Chen. TimePro: Efficient multivariate long-term time series forecasting with variable- and time-aware hyper-state. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 42096–42111. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/ma25p.html.

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*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

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

Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Junkai Lu, Jilin Hu, Chenjuan Guo, Christian S Jensen, and Bin Yang. A comprehensive survey of deep learning for multivariate time series forecasting: A channel strategy perspective. In *IJCAI*, 2026.

Minlan Shao, Zijian Zhang, Yili Wang, Yiwei Dai, Xu Shen, and Xin Wang. Hyperd: Hybrid periodicity decoupling framework for traffic forecasting. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 15689–15697. AAAI Press, 2026. doi: 10.1609/AAAI.V40I18.38599. URL https://doi.org/10.1609/aaai.v40i18.38599.

Peiwang Tang and Weitai Zhang. Unlocking the power of patch: Patch-based mlp for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 12640–12648, 2025.

Changyuan Tian, Zhicong Lu, Zequn Zhang, Heming Yang, Wei Cao, Zhi Guo, Xian Sun, and Li Jin. Hypermixer: Specializable hypergraph channel mixing for long-term multivariate time series forecasting. In Toby Walsh, Julie Shah, and Zico Kolter (eds.), *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pp. 20885–20893. AAAI Press, 2025. doi: 10.1609/AAAI.V39I19.34302. URL https://doi.org/10.1609/aaai.v39i19.34302.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

Lan Wu, Xuebin Wang, Chenglong Ge, Ruijuan Chu, and Linyu Wang. Exotimer: Leveraging large language models for time series forecasting with exogenous variables. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 26956–26964. AAAI Press, 2026a. doi: 10.1609/AAAI.V40I32.39908. URL https://doi.org/10.1609/aaai.v40i32.39908.

Xingjian Wu, Jianxin Jin, Wanghui Qiu, Peng Chen, Yang Shu, Bin Yang, and Guo Guo. Aurora: Towards universal generative multimodal time series forecasting. In *International Conference on Learning Representations*, volume 2026, pp. 103239–103277, 2026b.

Yingnan Yang, Qingling Zhu, and Jianyong Chen. Vcformer: Variable correlation transformer with inherent lagged correlation for multivariate time series forecasting. In Kate Larson (ed.), *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pp. 5335–5343. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/590. URL https://doi.org/10.24963/ijcai.2024/590. Main Track.

Jiexia Ye, Weiqi Zhang, Ziyue Li, Jia Li, Meng Zhao, and Fugee Tsung. Medualtime: A dual-adapter language model for medical time series-text multimodal learning. In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 7913–7921. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/880. URL https://doi.org/10.24963/ijcai.2025/880. Main Track.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are more effective learners in time series forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. Filternet: Harnessing frequency filters for time series forecasting. *Advances in Neural Information Processing Systems*, 37:55115–55140, 2024.

Guoqi Yu, Jing Zou, Xiaowei Hu, Angelica I Aviles-Rivero, Jing Qin, and Shujun Wang. Revitalizing multivariate time series forecasting: Learnable decomposition with inter-series dependencies and intra-series variations modeling. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=87CYNyCGOo.

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.

Bowen Zhang, Hongda Tian, Adam Berry, and A. Craig Roussac. Improving day-ahead grid carbon intensity forecasting by joint modeling of local-temporal and cross-variable dependencies across

different frequencies. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 39585–39593. AAAI Press, 2026a. doi: 10.1609/AAAI.V40I46.41310. URL https://doi.org/10.1609/aaai.v40i46.41310.

Fan Zhang, Shiming Fan, and Hua Wang. What if we let forecasting forget? a sparse bottleneck for cross-variable dependencies. *arXiv preprint arXiv:2605.08289*, 2026b.

Tianping Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.

Xu Zhang, Qitong Wang, Peng Wang, and Wei Wang. A lightweight sparse interaction network for time series forecasting. In Toby Walsh, Julie Shah, and Zico Kolter (eds.), *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pp. 13304–13312. AAAI Press, 2025. doi: 10.1609/AAAI.V39I12.33452. URL https://doi.org/10.1609/aaai.v39i12.33452.

Zhiwei Zhang, Xinyi Du, Xuanchi Guo, Weihao Wang, and Wenjuan Han. Emaformer: enhancing transformer through embedding armor for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 28636–28644, 2026c.

Lifan Zhao and Yanyan Shen. Rethinking channel dependence for multivariate time series forecasting: Learning from leading indicators. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=JiTVtCUOpS.

Ziyu Zhou, Yiming Huang, Yanyun Wang, Yuankai Wu, James Kwok, and Yuxuan Liang. Revitalizing canonical pre-alignment for irregular multivariate time series forecasting. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 29115–29123. AAAI Press, 2026. doi: 10.1609/AAAI.V40I34.40149. URL https://doi.org/10.1609/aaai.v40i34.40149.

A Appendix

B Dataset Descriptions

We evaluate TimeSpur on eight widely used multivariate time-series forecasting benchmarks, covering electricity, weather, solar-energy, and transportation systems. Their statistics are summarized in Table 3.

ETT. [1] The Electricity Transformer Temperature (ETT) dataset records the operating conditions of two electricity transformers from July 2016 to July 2018. Each subset contains the oil temperature and six power-load indicators. ETTh1 and ETTh2 are sampled hourly, whereas ETTm1 and ETTm2 provide finer observations at 15-minute intervals.

Electricity and Weather. Electricity[2] contains the hourly power consumption of 321 clients collected between 2012 and 2014. Weather[3] consists of 21 meteorological indicators, including temperature, humidity, pressure, and wind conditions, recorded every 10 minutes throughout 2020.

[1] https://github.com/zhouhaoyi/ETDataset

[2] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[3] https://www.bgc-jena.mpg.de/wetter/

Table 3: Statistics of the eight forecasting benchmarks. Length and Dim denote the total number of time points and variables, respectively. Split represents the chronological training, validation, and test ratio. Data sourced from iTransformer (Liu et al., 2024).

Dataset	Domain	Frequency	Length	Dim	Split
ETTh1	Electricity	1 hour	14,400	7	6:2:2
ETTh2	Electricity	1 hour	14,400	7	6:2:2
ETTm1	Electricity	15 min	57,600	7	6:2:2
ETTm2	Electricity	15 min	57,600	7	6:2:2
Electricity	Electricity	1 hour	26,304	321	7:1:2
Weather	Environment	10 min	52,696	21	7:1:2
Solar-Energy	Energy	10 min	52,560	137	6:2:2
Traffic	Transportation	1 hour	17,544	862	7:1:2

Solar-Energy and Traffic. Solar-Energy[4] records the power production of 137 photovoltaic plants in Alabama at 10-minute intervals. Traffic[5] contains hourly road-occupancy measurements collected by 862 sensors deployed across freeways in the San Francisco Bay Area.

Following standard forecasting protocols, all datasets are chronologically divided into training, validation, and test sets. We use a 6:2:2 split for the ETT and Solar-Energy datasets and a 7:1:2 split for Electricity, Weather, and Traffic. The look-back length is fixed to $L = 320$, and the prediction horizons are selected from $H \in \{96, 192, 336, 720\}$. For TimeSpur, the lagged candidate graph is constructed exclusively from the training split to prevent information leakage.

C Full Results

Table 4 reports the complete forecasting results on eight benchmarks under four prediction horizons $H \in \{96, 192, 336, 720\}$. The input length is fixed to $L = 320$ for all experiments, and performance is evaluated using MSE and MAE. TimeSpur achieves strong overall performance, with particularly competitive results on ETTh2, ETTm2, Electricity, and Solar-Energy, while remaining comparable to recent baselines on the other datasets. The best and second-best results are highlighted in red and blue, respectively.

D Additional Implementation Details

D.1 Data Processing and Evaluation Protocol

All datasets are divided chronologically, and the scaler is fitted exclusively on the training portion before being reused for validation and testing. No test statistic is involved in normalization, candidate construction, model selection, or early stopping. Unless otherwise stated, the look-back length is fixed to $L = 320$, and we evaluate four forecasting horizons $H \in \{96, 192, 336, 720\}$. The lagged candidate graph is fitted once from the training sequence and is stored with the checkpoint. Validation and test windows reuse this graph without recomputing correlations.

For a test set containing N windows, the reported metrics are

$$\mathrm{MSE} = \frac{1}{NHC}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{c=1}^{C}(\widehat{Y}_{nhc} - Y_{nhc})^2, \qquad \mathrm{MAE} = \frac{1}{NHC}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{c=1}^{C}|\widehat{Y}_{nhc} - Y_{nhc}|. \quad (11)$$

Both metrics are computed over the complete multivariate forecast, and lower values indicate better performance.

[4] https://www.nrel.gov/grid/solar-power-data.html

[5] https://pems.dot.ca.gov/

Table 4: Detailed long-term forecasting results on eight benchmarks. All experiments fix the look-back length to $L = 320$. Each *Avg* row reports the average over $H \in \{96, 192, 336, 720\}$. Lower MSE and MAE indicate better performance. The best result is shown in **red**, and the second-best result is shown in blue.

Models		TimeSpur (Ours)		XLinear (AAAI'2026)		EMAformer (AAAI'2026)		TimeFilter (ICML'2025)		PatchMLP (AAAI'2025)		iTransformer (ICLR'2024)		TimeMixer (ICLR'2024)		PatchTST (ICLR'2023)		DLinear (AAAI'2023)		FreTS (NIPS'2023)		LightTS (2022)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.375**	**0.398**	0.377	**0.398**	0.405	0.417	0.460	0.457	0.411	0.426	0.403	0.417	0.376	**0.398**	0.403	0.423	0.377	0.400	0.400	0.419	0.432	0.443
	192	**0.409**	**0.420**	0.412	0.421	0.443	0.442	0.439	0.443	0.443	0.445	0.443	0.444	0.412	**0.420**	0.479	0.478	0.410	**0.420**	0.436	0.441	0.474	0.472
	336	**0.437**	**0.439**	0.452	0.451	0.472	0.461	0.474	0.463	0.468	0.461	0.470	0.464	0.445	0.446	0.490	0.481	0.452	0.458	0.471	0.466	0.515	0.501
	720	**0.454**	**0.468**	0.455	0.470	0.515	0.499	0.670	0.564	0.525	0.508	0.632	0.573	0.488	0.489	0.641	0.559	0.477	0.497	0.537	0.532	0.592	0.567
	Avg	**0.419**	**0.431**	0.424	0.435	0.459	0.455	0.511	0.482	0.462	0.460	0.487	0.475	0.430	0.438	0.503	0.485	0.429	0.444	0.461	0.465	0.503	0.496
ETTh2	96	0.284	0.343	**0.276**	**0.338**	0.310	0.361	0.367	0.389	0.311	0.371	0.323	0.369	0.285	0.346	0.324	0.374	0.305	0.368	0.321	0.380	0.324	0.383
	192	**0.335**	**0.378**	0.344	0.382	0.398	0.413	0.372	0.404	0.381	0.413	0.421	0.422	0.358	0.391	0.378	0.416	0.407	0.435	0.396	0.429	0.438	0.455
	336	**0.361**	**0.399**	0.370	0.407	0.430	0.435	0.402	0.426	0.401	0.427	0.440	0.441	0.380	0.409	0.405	0.432	0.493	0.490	0.497	0.496	0.496	0.490
	720	**0.386**	**0.422**	0.404	0.436	0.443	0.455	0.465	0.464	0.426	0.451	0.459	0.466	0.410	0.439	0.484	0.484	0.762	0.620	0.766	0.628	1.151	0.749
	Avg	**0.342**	**0.386**	0.349	0.391	0.395	0.416	0.402	0.421	0.380	0.416	0.411	0.425	0.358	0.396	0.398	0.427	0.492	0.478	0.495	0.483	0.602	0.519
ETTm1	96	0.291	0.343	**0.285**	**0.340**	0.303	0.353	0.355	0.373	0.305	0.357	0.327	0.371	0.307	0.352	0.321	0.361	0.299	0.344	0.308	0.353	0.313	0.363
	192	0.331	0.367	**0.324**	**0.364**	0.346	0.379	0.376	0.396	0.351	0.384	0.357	0.388	0.332	0.373	0.342	0.385	0.337	0.370	0.346	0.377	0.350	0.387
	336	0.367	0.390	**0.354**	**0.385**	0.384	0.401	0.401	0.413	0.387	0.403	0.393	0.412	0.362	0.390	0.386	0.410	0.369	**0.385**	0.382	0.399	0.399	0.418
	720	0.425	**0.418**	**0.412**	**0.418**	0.443	0.435	0.465	0.449	0.454	0.441	0.467	0.452	0.422	0.426	0.439	0.441	0.424	0.419	0.440	0.433	0.489	0.479
	Avg	0.354	0.379	**0.344**	**0.377**	0.369	0.392	0.399	0.408	0.374	0.396	0.386	0.406	0.356	0.385	0.372	0.399	0.357	0.380	0.369	0.391	0.388	0.412
ETTm2	96	**0.167**	**0.256**	0.170	0.257	0.178	0.261	0.174	0.261	0.179	0.265	0.187	0.271	0.169	0.258	0.181	0.270	0.172	0.270	0.175	0.263	0.187	0.286
	192	**0.220**	**0.288**	0.225	0.296	0.242	0.307	0.229	0.298	0.232	0.301	0.251	0.314	0.240	0.308	0.241	0.311	0.239	0.322	0.248	0.314	0.311	0.384
	336	**0.275**	**0.327**	0.277	0.331	0.296	0.341	0.312	0.351	0.285	0.334	0.319	0.355	0.282	0.334	0.303	0.348	0.305	0.367	0.321	0.364	0.399	0.418
	720	0.369	**0.368**	**0.362**	0.384	0.374	0.387	0.396	0.408	0.380	0.394	0.413	0.412	0.369	0.391	0.385	0.399	0.412	0.432	0.393	0.414	0.489	0.479
	Avg	**0.258**	**0.310**	0.259	0.317	0.273	0.324	0.278	0.330	0.269	0.324	0.293	0.338	0.265	0.323	0.278	0.332	0.282	0.348	0.284	0.339	0.347	0.392
Electricity	96	**0.130**	**0.225**	0.138	0.238	0.133	0.227	0.137	0.233	0.165	0.271	0.145	0.244	0.141	0.238	0.142	0.245	0.143	0.242	0.144	0.245	0.171	0.285
	192	**0.147**	**0.241**	**0.147**	0.242	0.153	0.244	0.153	0.246	0.183	0.288	0.165	0.263	0.158	0.254	0.159	0.259	0.157	0.255	0.156	0.253	0.192	0.305
	336	**0.158**	**0.258**	0.165	0.259	0.168	0.263	0.169	0.263	0.201	0.304	0.184	0.282	0.175	0.270	0.176	0.276	0.172	0.272	0.173	0.272	0.214	0.324
	720	**0.188**	0.294	0.204	0.295	0.193	**0.285**	0.208	0.296	0.250	0.343	0.227	0.317	0.217	0.307	0.215	0.308	0.207	0.305	0.210	0.308	0.251	0.349
	Avg	**0.156**	**0.255**	0.164	0.259	0.162	0.255	0.167	0.260	0.200	0.302	0.180	0.277	0.173	0.267	0.173	0.272	0.170	0.269	0.171	0.270	0.207	0.316
Weather	96	**0.146**	**0.199**	0.150	0.201	0.154	0.208	0.156	0.209	0.153	0.206	0.160	0.209	0.150	**0.199**	0.155	0.204	0.177	0.242	0.156	0.213	0.152	0.212
	192	**0.189**	**0.240**	0.191	0.241	0.195	0.245	0.195	0.246	0.197	0.245	0.207	0.251	0.193	0.244	0.201	0.249	0.217	0.276	0.197	0.253	0.197	0.255
	336	**0.240**	0.288	**0.240**	**0.280**	0.244	**0.280**	0.251	0.286	0.247	0.282	0.256	0.289	0.244	0.283	0.255	0.289	0.262	0.312	0.246	0.293	0.249	0.299
	720	**0.313**	0.342	0.320	**0.334**	0.323	0.336	0.322	0.339	0.324	**0.334**	0.325	0.338	0.319	0.336	0.333	0.340	0.326	0.367	0.319	0.347	0.323	0.357
	Avg	**0.222**	0.267	0.225	**0.264**	0.229	0.267	0.231	0.270	0.230	0.267	0.237	0.272	0.227	0.266	0.236	0.271	0.246	0.299	0.230	0.277	0.230	0.281
Solar-Energy	96	**0.177**	**0.239**	–	–	0.192	0.247	0.182	0.240	0.211	0.280	0.198	0.253	0.199	0.257	0.195	0.249	0.223	0.293	0.190	0.248	0.196	0.267
	192	**0.194**	**0.249**	–	–	0.205	0.257	0.199	0.255	0.245	0.285	0.222	0.276	0.221	0.274	0.208	0.257	0.251	0.311	0.209	0.266	0.211	0.280
	336	**0.195**	**0.254**	–	–	0.208	0.262	0.207	0.262	0.271	0.302	0.251	0.293	0.230	0.284	0.220	0.269	0.272	0.327	0.222	0.274	0.229	0.299
	720	**0.200**	**0.256**	–	–	0.210	0.274	0.213	0.264	0.272	0.303	0.256	0.299	0.224	0.290	0.220	0.272	0.274	0.330	0.228	0.278	0.232	0.304
	Avg	**0.192**	**0.250**	–	–	0.204	0.260	0.200	0.255	0.250	0.293	0.232	0.280	0.219	0.276	0.211	0.262	0.255	0.315	0.212	0.267	0.217	0.288
Traffic	96	0.410	0.281	0.391	**0.274**	**0.387**	0.283	0.397	**0.274**	0.486	0.361	0.422	0.310	0.421	0.297	0.403	0.285	0.422	0.296	0.417	0.301	0.493	0.367
	192	0.426	0.291	0.412	0.286	0.415	0.296	**0.411**	**0.278**	0.515	0.378	0.449	0.327	0.441	0.311	0.419	0.292	0.435	0.302	0.447	0.299	0.511	0.373
	336	0.443	0.298	0.432	0.296	0.429	0.296	**0.423**	**0.284**	0.533	0.386	0.469	0.339	0.425	0.317	0.432	0.298	0.447	0.308	0.468	0.311	0.521	0.378
	720	0.469	0.319	0.455	0.311	0.451	0.310	**0.449**	**0.300**	0.564	0.401	0.512	0.364	0.489	0.340	0.459	0.313	0.476	0.327	0.512	0.340	0.504	0.360
	Avg	0.437	0.297	0.423	0.292	0.421	0.296	**0.420**	**0.284**	0.525	0.382	0.463	0.335	0.444	0.316	0.428	0.297	0.445	0.308	0.461	0.313	0.507	0.370

D.2 Optimization and Model Selection

TimeSpur is implemented in PyTorch and trained with Adam using the MSE forecasting objective and the equally weighted utility-calibration loss in Eq. 10. The maximum number of epochs is 30. We halve the learning rate after each epoch and apply early stopping with patience 3 according to validation MSE. The checkpoint with the lowest validation loss is used once for test evaluation. All main experiments are run on one NVIDIA RTX 5080 GPU with 16 GB memory. We fix the random seed to 2025 for model training.

D.3 Model Size across Datasets

The parameter count changes with C, H, and D, but not with the number of activated edges. Table 5 reports the exact model sizes for $H = 96$. Even on Traffic with 862 variables, TimeSpur remains below 0.3M parameters because candidate indices and signed graph weights are non-learnable buffers and the router uses low-rank source/target projections.

Table 5: TimeSpur model size for $H = 96$. K_{eff} is the actual number of non-self candidate sources per target.

Dataset	C	K_{eff}	D	**Params**
ETTh1	7	3	32	86,904
ETTh2	7	6	16	78,504
ETTm1	7	6	48	95,304
ETTm2	7	6	256	204,504
Electricity	321	64	32	167,288
Weather	21	16	128	140,328
Solar-Energy	137	128	256	245,016
Traffic	862	256	16	284,016

E Training and Inference Procedure

Offline preparation. Given the chronological training split, we standardize the adjacent slices $Z_{1:T-1}$ and $Z_{2:T}$, compute the directed lagged relation matrix in Eq. 4, remove self-connections, and retain the K largest absolute entries for every target. The selected signed values are row-normalized to obtain (N, Ω). This step is executed once and does not use validation or test observations.

Mini-batch training. For every batch, TimeSpur first produces a complete-input forecast and the state representation H. One source variable is sampled independently for each instance. We remove its normalized history, mean, and variance and run one additional masked forecast. Comparing the complete and masked forecasts against the same observed future gives one measured utility row covering all targets. The utility probe predicts the corresponding row from the detached historical states. Finally, the forecasting loss and utility-calibration loss are summed, back-propagated, and used to update the model. Thus, the procedure requires two forecasting passes per training batch rather than enumerating C masked inputs.

Inference. At test time, TimeSpur computes the independent forecast and current states, scores only the CK candidate edges, activates scores above zero, and adds the resulting sparse residual to the base prediction. Neither stochastic masking nor future targets are used. The test-time computation is therefore a single deterministic forward pass.

F Complexity Analysis

Let T be the length of the training split, R_u the utility rank, and K the number of candidates per target. Constructing the lagged relation matrix once costs $\mathcal{O}(TC^2)$ time and $\mathcal{O}(C^2)$ temporary memory. Only N and Ω are retained, requiring $\mathcal{O}(CK)$ storage. This offline cost is not repeated during mini-batch training or inference.

For a batch of size B, the state encoder costs $\mathcal{O}(BCLD)$. The low-rank utility projections cost $\mathcal{O}(BCDR_u)$, candidate scoring costs $\mathcal{O}(BCKR_u)$, and sparse aggregation costs $\mathcal{O}(BCKD)$. The router stores $\mathcal{O}(BCK + BCD)$ activations and never materializes a $B \times C \times C$ matrix. In contrast, dense all-variable routing requires $\mathcal{O}(BC^2(R_u + D))$ operations and $\mathcal{O}(BC^2)$ edge memory. Consequently, the cross-variable component scales linearly in K rather than C when $K \ll C$.

Paired masking adds one forecasting pass during training, but it supplies the utility of one sampled source to all C targets simultaneously. Exhaustive source masking would instead require C additional passes. At inference, paired masking is removed entirely, which is consistent with the latency and memory results in Fig. 3.

G Additional Analysis of Sparse Routing

G.1 Effective Sparsity on the Test Set

We quantify the executed graph, rather than only reporting the configured candidate count. Using the Electricity $H = 96$ checkpoint, we scan all $5,165$ test windows and record every binary gate. The offline graph contains $K = 64$ candidates per target, corresponding to 20.0% of the complete non-self graph. At the default threshold, TimeSpur activates 57.79 candidates on average, or 18.06% of all possible directed edges. The mean target-specific routing budget ranges from 19.24 to 63.89, and 4.67% of candidate decisions switch between adjacent windows. Figure 6 shows that the effective budget varies across both targets and historical volatility regimes rather than remaining a fixed top-K graph.

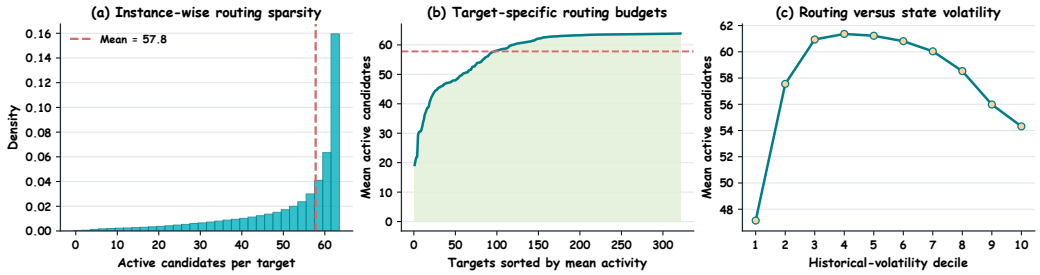
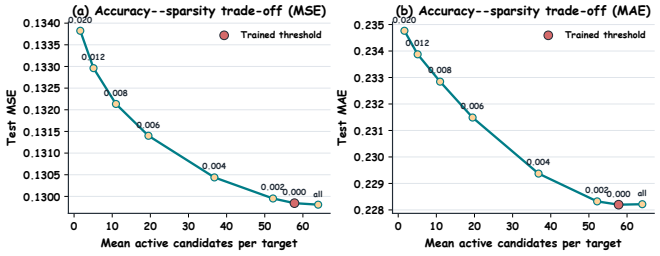


Figure 6: **Executed routing statistics on Electricity.** (a) Distribution of active candidates over all target–window pairs. (b) Mean active candidates for each target. (c) Mean routing budget across deciles of recent historical volatility.

G.2 Inference Threshold Sensitivity

We keep the trained checkpoint fixed and vary only the utility threshold used by the inference gate. This isolates the accuracy–sparsity trade-off from retraining or hyperparameter selection. As shown in Fig. 7, the default threshold 0 filters about 9.7% of the offline candidates while matching the accuracy of the all-active graph. Increasing the threshold to 0.006 reduces the executed graph to 6.09% of all possible edges, with MSE changing from 0.1298 to 0.1314. Even at threshold 0.012, only 5.10 candidates are active per target and performance degrades gradually. This indicates that TimeSpur exposes a controllable inference-time trade-off when a stricter computation or communication budget is required.



Selected operating points

δ	Active K	Density	MSE
All	64.00	20.00%	0.1298
0.000	57.79	18.06%	0.1298
0.004	36.79	11.50%	0.1304
0.006	19.50	6.09%	0.1314
0.012	5.10	1.59%	0.1330
0.020	1.63	0.51%	0.1338

Figure 7: **Inference-time accuracy–sparsity trade-off on Electricity.** The checkpoint is fixed, and only the utility threshold δ is changed. The red point denotes the default threshold 0; the table lists representative operating points.

G.3 Held-Out Utility-Sign Diagnostic

We additionally check whether the binary decisions made from history agree with paired-masking utility measured using the held-out future. We uniformly sample 512 Electricity test windows and cycle the masked source across all variables, producing $32,106$ candidate-edge instances. This future-dependent measurement is used only for post-hoc diagnosis and is never fed to the inference

router. As reported in Table 6, edges predicted to be active have positive measured utility in 52.33% of instances, compared with 26.53% for suppressed edges. Their mean measured utilities also have opposite signs. The result supports the gate at the group level, while the small magnitudes emphasize that a single-window counterfactual estimate is noisy and should not be interpreted as a causal effect.

Table 6: Post-hoc utility-sign diagnostic on Electricity. Measured utility is the normalized error increase caused by masking the source.

Predicted gate	**Instances**	**Mean measured utility**	$u^* > 0$
Active	29,027	$+1.43 \times 10^{-4}$	52.33%
Inactive	3,079	-4.08×10^{-4}	26.53%

H Qualitative Evaluation Protocol

The state-dependent heatmaps in Fig. 5 are selected without consulting future targets. Targets and candidate edges are ranked by temporal score variation, activation balance, and sign-switch frequency; for each selected target, we display a contiguous 96-window interval with high utility variation. Each heatmap column therefore corresponds to one observed history state, and a positive entry is exactly an edge that would be active at inference.

For Fig. 4, the pre-routing prediction is recovered exactly by subtracting the denormalized sparse residual from the final TimeSpur forecast. It retains decomposition, variable-specific calibration, the temporal-state correction, and calendar covariates. Representative cases are chosen only for visualization by ranking relative MSE reduction after excluding nearly constant or already trivial samples; selected examples use distinct targets and are separated by at least 96 test hours. This selection does not affect training, model selection, or any aggregate result. Curves are shown in the original target scale, whereas the annotated MSE values follow the standardized evaluation space used by the benchmark protocol.

I Robustness across random seeds.

To further assess the robustness of TimeSpur to random initialization, we repeat the experiments with three random seeds (2024, 2025, and 2026) on ETTh2, ETTm2, Weather, and Electricity, covering four prediction horizons $H \in \{96, 192, 336, 720\}$. As shown in Table 7, TimeSpur exhibits stable performance across different random seeds, with only small variations in both MSE and MAE. Moreover, after averaging the three seeds for each dataset–horizon pair, TimeSpur achieves a lower overall MSE than XLinear and obtains lower MSE on 13 out of 16 forecasting settings. A two-sided Wilcoxon signed-rank test over the 16 paired settings confirms that the MSE improvement is statistically significant ($W = 25.0$, $p = 0.025$). A paired t-test leads to the same conclusion ($p = 0.043$). These results indicate that TimeSpur's performance advantage is consistent across different random seeds and does not stem from any specific favorable initialization.

J Visualization of prediction results

Figures 8–11 present representative forecasting results on four datasets with different prediction lengths. For each dataset, all methods use the same historical input and predict the same target sequence. Compared with the baseline models, TimeSpur generally produces predictions that are closer to the ground truth across both short- and long-term forecasting settings. These qualitative results further demonstrate the effectiveness of TimeSpur under diverse temporal patterns and forecasting horizons.

K Limitations

TimeSpur deliberately separates candidate recall from dynamic edge selection, but its router cannot recover a useful relation that is absent from the offline candidate set. The current graph uses one-step linear lagged correlation; nonlinear or longer-delay relations may therefore require a richer

Models		TimeSpur (Ours)						XLinear (AAAI'2026)					
Seed		2024		2025		2026		2024		2025		2026	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh2	96	0.284	0.344	0.284	0.343	0.283	0.342	0.276	0.337	0.276	0.338	0.278	0.339
	192	0.340	0.378	0.335	0.378	0.342	0.381	0.350	0.385	0.344	0.382	0.341	0.381
	336	0.365	0.399	0.361	0.399	0.367	0.401	0.369	0.407	0.370	0.407	0.369	0.406
	720	0.386	0.423	0.386	0.422	0.388	0.423	0.403	0.436	0.404	0.436	0.402	0.435
	Avg	0.344	0.386	0.342	0.386	0.345	0.387	0.349	0.391	0.348	0.391	0.348	0.390
ETTm2	96	0.169	0.257	0.167	0.256	0.175	0.260	0.170	0.256	0.171	0.257	0.169	0.257
	192	0.223	0.294	0.220	0.288	0.227	0.298	0.224	0.295	0.225	0.296	0.227	0.298
	336	0.275	0.328	0.275	0.327	0.275	0.328	0.279	0.332	0.277	0.331	0.277	0.330
	720	0.371	0.386	0.369	0.368	0.367	0.383	0.366	0.386	0.362	0.385	0.365	0.385
	Avg	0.260	0.316	0.258	0.310	0.261	0.317	0.260	0.317	0.259	0.317	0.259	0.317
Weather	96	0.146	0.203	0.146	0.199	0.148	0.206	0.145	0.196	0.150	0.201	0.147	0.199
	192	0.189	0.246	0.189	0.240	0.190	0.246	0.192	0.242	0.191	0.241	0.189	0.239
	336	0.241	0.288	0.240	0.288	0.240	0.288	0.240	0.278	0.240	0.280	0.245	0.281
	720	0.314	0.343	0.313	0.342	0.313	0.341	0.319	0.331	0.320	0.334	0.320	0.334
	Avg	0.223	0.270	0.222	0.267	0.223	0.270	0.224	0.262	0.225	0.264	0.225	0.263
Electricity	96	0.130	0.228	0.130	0.225	0.130	0.228	0.129	0.230	0.138	0.238	0.133	0.231
	192	0.148	0.247	0.147	0.241	0.148	0.247	0.151	0.248	0.147	0.242	0.147	0.242
	336	0.160	0.263	0.158	0.258	0.161	0.264	0.166	0.259	0.165	0.259	0.166	0.259
	720	0.188	0.293	0.188	0.294	0.188	0.293	0.204	0.293	0.204	0.295	0.201	0.291
	Avg	0.156	0.258	0.156	0.255	0.157	0.258	0.162	0.257	0.164	0.258	0.162	0.256

Table 7: Detailed forecasting results across three random seeds on four benchmarks. All results are reported for prediction lengths $H \in \{96, 192, 336, 720\}$ and random seeds $\{2024, 2025, 2026\}$. Each *Avg* row reports the average over the four prediction lengths for the corresponding seed. Lower MSE and MAE indicate better performance.

proposal mechanism. The $\mathcal{O}(TC^2)$ graph construction is performed only once and is practical for the benchmarks considered here, but it can become expensive for extremely large variable sets. Approximate nearest-neighbor search or blockwise correlation would reduce this offline cost.

Paired masking measures conditional predictive contribution under the trained forecasting model, not a causal intervention on the underlying physical system. Its labels can also be noisy when several variables carry redundant information or when the realized future is atypical. Finally, the current implementation samples one source per training instance. Sampling multiple sources could reduce label variance but would increase training cost. These limitations motivate future work on adaptive candidate refresh, nonlinear proposal scores, and uncertainty-aware utility supervision.

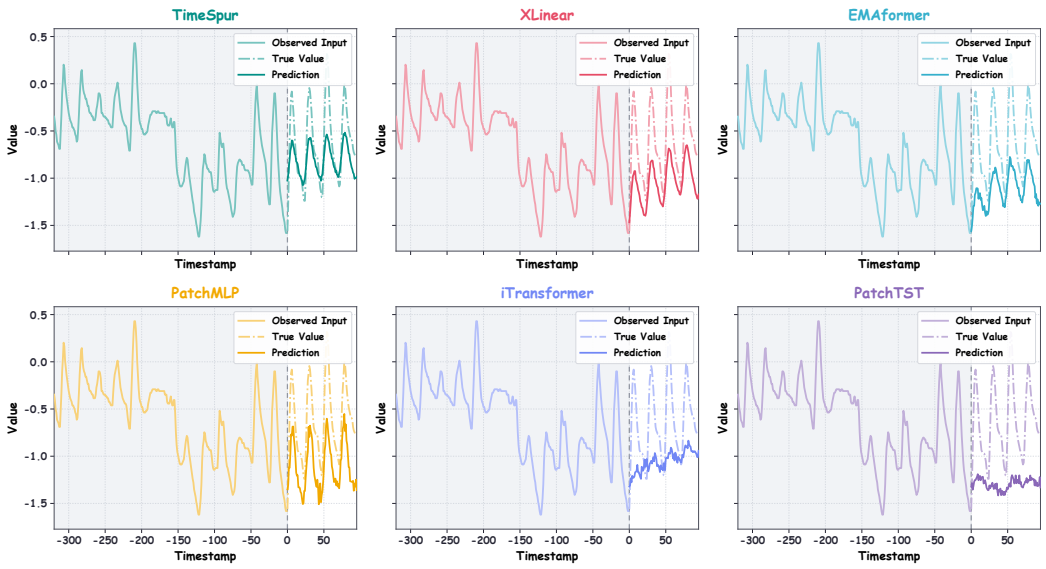


Figure 8: **Qualitative forecasting comparison on ETTh2** ($H = 96$)**.** All models use the same historical window with $L = 320$, target variable, and ground-truth future. The vertical dashed line marks the beginning of the forecasting horizon.

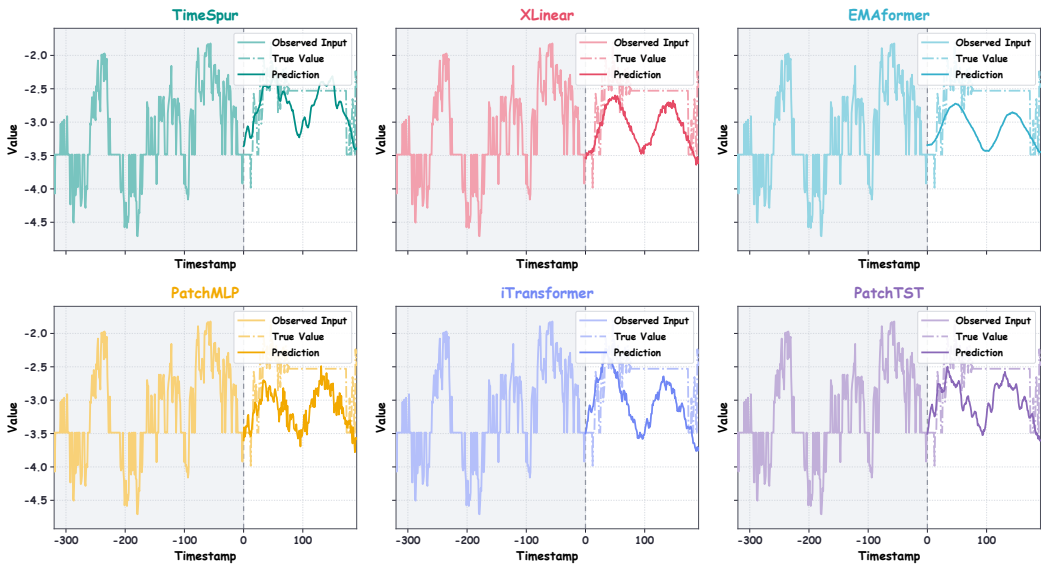


Figure 9: **Qualitative forecasting comparison on ETTm2** ($H = 192$)**.** Predictions from six models are presented using an identical input window of length $L = 320$ and the same target sequence. The vertical dashed line separates the observed history from the future horizon.

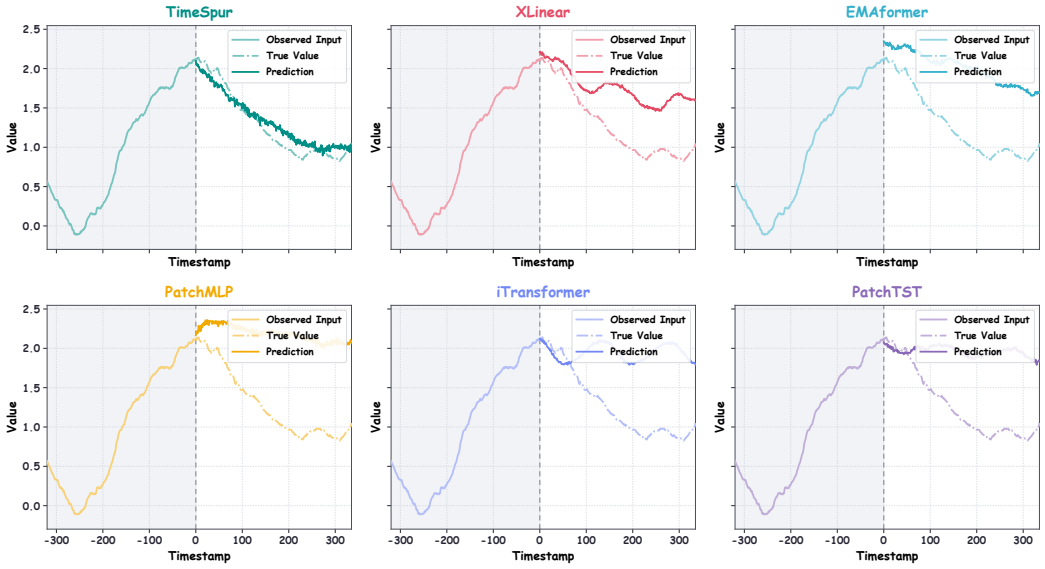


Figure 10: **Qualitative forecasting comparison on Weather** ($H = 336$)**.** All methods forecast the same target variable from a shared historical input of length $L = 320$. The vertical dashed line indicates the forecasting boundary.

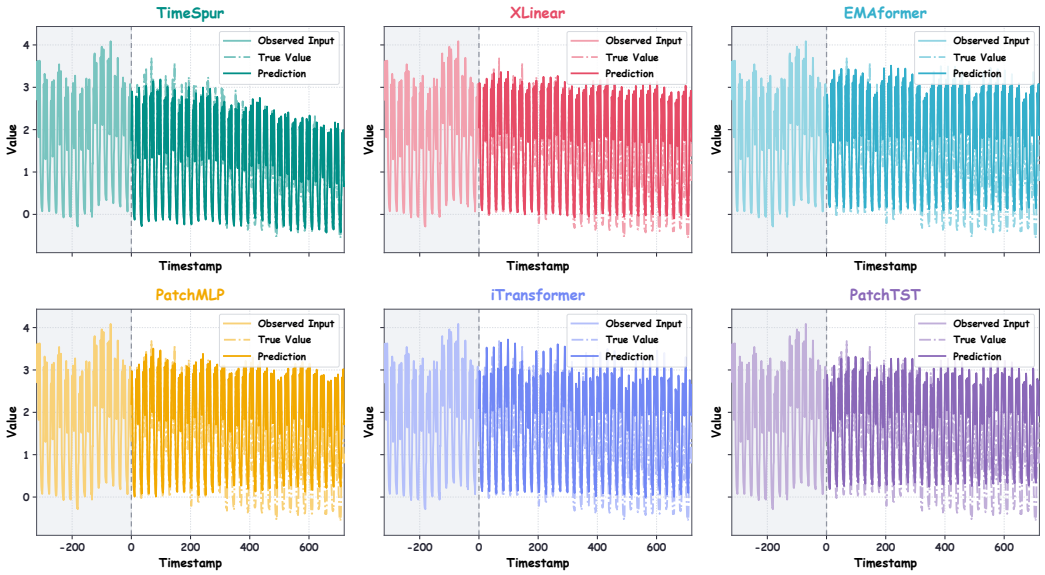


Figure 11: **Qualitative forecasting comparison on Electricity** ($H = 720$)**.** The six models are evaluated on the same input window, target variable, and future sequence, with the look-back length fixed to $L = 320$. The vertical dashed line denotes the start of long-horizon forecasting.