TIMERELAY: RELATION-AWARE RECURRENT VARIABLE MODELING FOR MULTIVARIATE TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

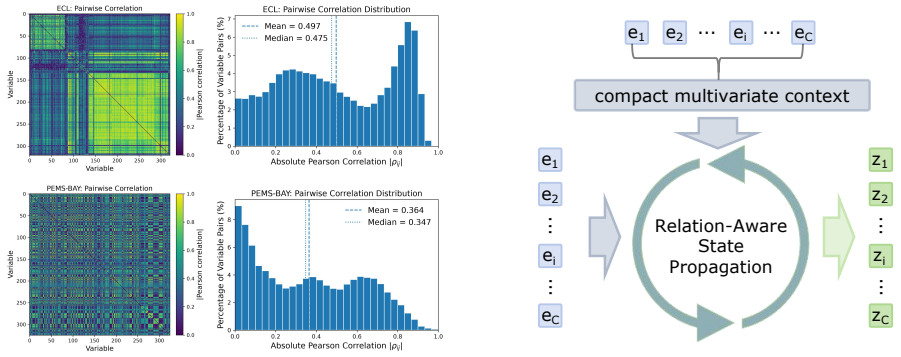
Multivariate time series forecasting requires effective modeling of temporal dynamics within individual variables and predictive dependencies across variables. A representative class of cross-variable modeling approaches explicitly constructs variable-to-variable interactions, resulting in a candidate interaction space that grows quadratically with the number of variables. At the same time, the predictive relevance of different variable pairs can be highly non-uniform. Motivated by these observations, we propose **TimeRelay**, a relation-aware recurrent framework that models cross-variable dependencies from the perspective of predictive information propagation rather than explicitly materializing dense pairwise interactions. TimeRelay first constructs variable-wise temporal representations from historical observations and then applies **Relation-Aware State Propagation (RASP)**. For each variable, RASP constructs a compact multivariate context and uses the variable–context relation to adaptively control recurrent state updates. RASP therefore propagates predictive information across variables through evolving recurrent states without explicitly constructing a dense variable-to-variable interaction matrix. Its affine recurrent form enables work-efficient parallel evaluation with linear arithmetic work in the number of variables and logarithmic dependency depth. Extensive experiments on diverse multivariate benchmarks demonstrate the effectiveness, efficiency, and scalability of TimeRelay.

1 INTRODUCTION

Multivariate time series forecasting aims to predict future observations from the historical measurements of multiple variables and plays an important role in applications such as traffic forecasting (Yin et al., 2022), energy forecasting (Qian et al., 2019), and weather prediction (Wu et al., 2023b; Bi et al., 2023). Compared with univariate forecasting, an additional challenge lies in effectively exploiting information shared across variables. The future evolution of one variable may benefit from the historical behavior of others, making cross-variable modeling an important component of multivariate forecasting. Many channel-dependent forecasting models therefore seek to capture dependencies among variables in addition to modeling temporal dynamics within each individual series (Liu et al., 2024; Zhang & Yan, 2023; Lin et al., 2025).

A representative approach to cross-variable modeling is to explicitly construct interactions among variable representations, for example through attention or other pairwise mixing mechanisms (Liu et al., 2024; Zhang & Yan, 2023; Lin et al., 2025; Qiu et al., 2025; Hu et al., 2025). Such designs provide flexible information exchange and have achieved strong forecasting performance. Nevertheless, when dense pairwise interactions are explicitly materialized, the candidate interaction space grows quadratically with the number of variables, making cross-variable modeling increasingly expensive in high-dimensional multivariate systems. Meanwhile, relationships among variables can be highly heterogeneous. As illustrated in Figure 1(a), pairwise correlations on representative datasets exhibit markedly non-uniform strengths, providing a data-level illustration of this heterogeneity. Some variable pairs may provide complementary information for forecasting, whereas others may be weakly related, redundant, or only marginally useful. Although dense interaction mechanisms can learn to assign different importance to different variable pairs, they still operate over the full candidate interaction space. This raises a more fundamental question: *is it necessary to explicitly*

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.



(a) Empirical observations on pairwise interactions

(b) From pairwise interaction to state-mediated propagation

Figure 1: Motivation for **TimeRelay**. (a) Pairwise Pearson correlations on ECL and PEMS-BAY illustrate the quadratically growing interaction space and heterogeneous relationships among variables. (b) TimeRelay propagates predictive information across variables through relation-aware recurrent state transitions conditioned on compact multivariate context.

materialize the entire pairwise interaction space to effectively communicate predictive information across variables?

We distinguish between identifying cross-variable dependencies and propagating predictive information across variables. Effective forecasting does not necessarily require explicitly recovering a complete variable-to-variable interaction structure; instead, each variable needs access to useful context from the multivariate system. Motivated by this perspective, we propose **TimeRelay**, a relation-aware recurrent framework that shifts cross-variable modeling from explicit pairwise interactions to state-mediated information propagation, as illustrated in Figure 1(b).

TimeRelay first constructs variable-wise temporal representations using a linear base embedding with multi-scale residual enhancement, and then models cross-variable dependencies through **Relation-Aware State Propagation (RASP)**. RASP builds a compact multivariate context for each variable and uses the variable–context relation to modulate recurrent state updates. In this way, predictive information is propagated across variables through recurrent states without explicitly constructing a dense pairwise interaction matrix. To reduce the long dependency path of sequential recurrent evaluation, we formulate the RASP transition as an affine recurrence whose transformations can be composed associatively. This enables work-efficient parallel evaluation with linear arithmetic work in the number of variables and logarithmic ideal dependency depth. We further employ bidirectional propagation to reduce the asymmetry introduced by a single computational traversal, without assuming that the traversal reflects an intrinsic semantic ordering among variables.

We evaluate TimeRelay on diverse multivariate forecasting benchmarks with different domains and variable dimensionalities. Beyond forecasting accuracy, we analyze relation-aware conditioning, state-mediated propagation, variable traversal sensitivity, and computational scalability. The results demonstrate strong forecasting performance and favorable efficiency, supporting relation-aware state-mediated propagation as an alternative to explicitly materializing dense pairwise interactions.

Our main contributions are summarized as follows:

- We revisit cross-variable modeling from the perspective of *predictive information propagation*, and explore state-mediated propagation as an alternative to explicitly materializing dense pairwise interactions.
- We introduce **Relation-Aware State Propagation (RASP)**, which conditions recurrent state updates on the relation between each variable and a compact multivariate context. Its affine state transitions can be composed associatively, enabling work-efficient bidirectional propagation with linear arithmetic work in the number of variables and logarithmic dependency depth.
- Extensive experiments demonstrate strong forecasting performance and validate the effectiveness, efficiency, scalability, and traversal robustness of TimeRelay.

2 Related Work

Multivariate Time Series Forecasting. Existing forecasting methods can be broadly categorized into variable-dependent temporal modeling, variable-independent forecasting, and explicit cross-variable modeling. Representative approaches include Transformer-, convolution-, MLP-, graph-, and state-space-based architectures (Zhou et al., 2021; Wu et al., 2021; 2023a; Nie et al., 2023; Liu et al., 2024; Luo & Wang, 2024; Ekambaram et al., 2023; Wang et al., 2025b; Hu et al., 2025). In particular, recent methods increasingly focus on exploiting information shared across variables through attention, structured interaction, or selective mixing (Zhang & Yan, 2023; Liu et al., 2024; Lin et al., 2025; Qiu et al., 2025; Hu et al., 2025). TimeRelay differs from these approaches by propagating cross-variable information through recurrent states rather than explicitly materializing dense variable-pair interactions.

Recurrent and State-Space Modeling. Recurrent models such as LSTM (Hochreiter & Schmidhuber, 1997) and GRU (Cho et al., 2014) propagate information through compact hidden states, while recent state-space models such as Mamba (Gu & Dao, 2024) introduce selective state transitions for sequence modeling. Recurrent and state-space mechanisms have also been explored for multivariate time series forecasting (Wang et al., 2025b; Ma et al., 2025). TimeRelay differs by explicitly conditioning cross-variable state transitions on the relation between each variable and a compact multivariate context. The resulting affine recurrent form supports work-efficient parallel evaluation. A more detailed discussion of related works is provided in Appendix A.

3 Problem Formulation

Consider a multivariate time series with C variables. Given a historical observation window of length L, we denote the input as $\mathbf{X} \in \mathbb{R}^{L\times C}$, where each column $\mathbf{x}_i \in \mathbb{R}^L$ contains the historical observations of the i-th variable. The goal of multivariate time series forecasting is to predict the future values of all variables over a horizon of length H, denoted as $\mathbf{Y} \in \mathbb{R}^{H\times C}$.

Formally, a forecasting model f_θ parameterized by θ learns the mapping

$$\widehat{\mathbf{Y}} = f_\theta(\mathbf{X}), \tag{1}$$

where $\widehat{\mathbf{Y}} \in \mathbb{R}^{H\times C}$ denotes the predicted future sequence.

In this work, we focus on the cross-variable modeling component of f_θ. We first encode each variable independently along the temporal dimension, obtaining

$$\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_C] \in \mathbb{R}^{C\times D}, \tag{2}$$

where $\mathbf{e}_i \in \mathbb{R}^D$ denotes the temporal representation of the i-th variable. Given these variable-wise representations, our objective is to enable effective propagation of predictive information across variables without explicitly materializing a dense pairwise interaction structure.

4 Method

4.1 Overview

TimeRelay decomposes multivariate forecasting into variable-wise temporal representation learning and relation-aware cross-variable state propagation. As illustrated in Figure 2, given $\mathbf{X} \in \mathbb{R}^{L\times C}$, TimeRelay first independently encodes each variable into a temporal representation, producing $\mathbf{E} \in \mathbb{R}^{C\times D}$. TimeRelay then applies **Relation-Aware State Propagation (RASP)** to model cross-variable information. Instead of explicitly constructing a dense $C \times C$ interaction structure, RASP relates each variable to a compact multivariate context and uses their relation to adaptively control recurrent state propagation. Bidirectional propagation is employed to reduce the asymmetry introduced by a single computational traversal, yielding relation-enhanced representations $\mathbf{Z} \in \mathbb{R}^{C\times D}$.

Finally, a prediction head maps the relation-enhanced representations to the future sequence. The overall computation is summarized as

$$\mathbf{E} = \mathcal{T}(\mathbf{X}), \qquad \mathbf{Z} = \mathcal{R}(\mathbf{E}), \qquad \widehat{\mathbf{Y}} = \mathcal{P}(\mathbf{Z}), \tag{3}$$

where $\mathcal{T}$ denotes the temporal embedding, $\mathcal{R}$ denotes RASP, and $\mathcal{P}$ denotes the prediction head.

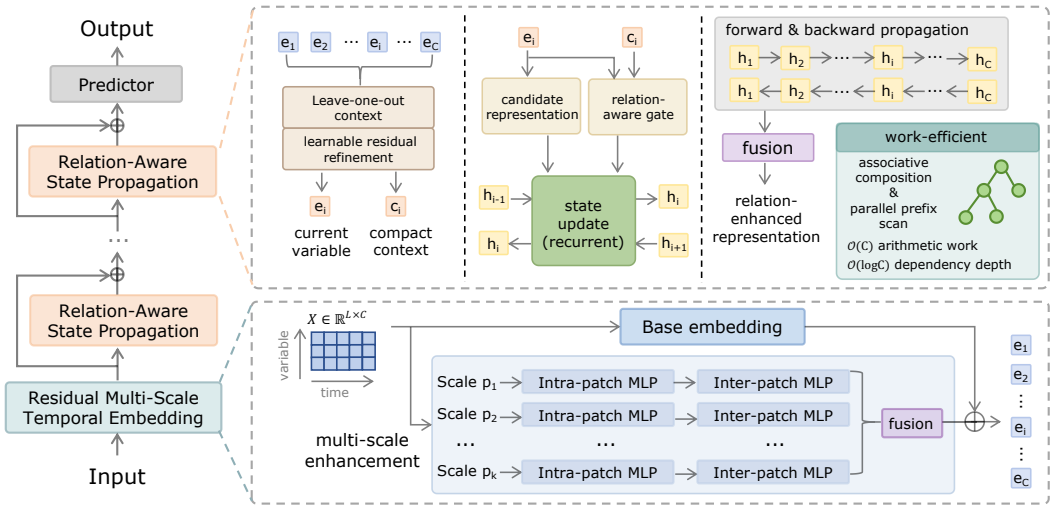


Figure 2: Overall architecture of **TimeRelay**. Variable-wise temporal representations are first constructed by residual multi-scale temporal embedding and then enhanced by Relation-Aware State Propagation (RASP) with work-efficient bidirectional recurrent propagation.

4.2 Residual Multi-Scale Temporal Embedding

Before cross-variable modeling, we independently encode the historical observations of each variable into a compact temporal representation. Given $\mathbf{x}_i \in \mathbb{R}^L$, we first apply a linear projection to obtain a base embedding,

$$\mathbf{e}_i^{\text{base}} = \mathbf{W}_b \mathbf{x}_i + \mathbf{b}_b, \qquad \mathbf{e}_i^{\text{base}} \in \mathbb{R}^D. \tag{4}$$

This provides a direct representation of the complete historical window.

To complement the base embedding with multi-scale temporal patterns, we extract lightweight features from several temporal resolutions $p \in P$. At each resolution, the historical sequence is partitioned into temporal patches of size p and processed by lightweight intra- and inter-patch MLPs, yielding a scale-specific feature $\mathbf{f}_i^{(p)}$. The resulting multi-scale features are fused into a residual temporal correction,

$$\Delta\mathbf{e}_i = \mathcal{F}_{\text{ms}}\left(\text{Concat}\left[\left\{\mathbf{f}_i^{(p)}\right\}_{p\in P}\right]\right), \qquad \Delta\mathbf{e}_i \in \mathbb{R}^D, \tag{5}$$

where $\mathcal{F}_{\text{ms}}$ denotes a lightweight fusion network and P denotes the set of temporal patch sizes.

The final temporal representation is obtained through residual refinement,

$$\mathbf{e}_i = \mathbf{e}_i^{\text{base}} + \Delta\mathbf{e}_i. \tag{6}$$

Collecting all variable representations gives

$$\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_C] \in \mathbb{R}^{C\times D}. \tag{7}$$

All temporal encoding operations are performed independently for each variable; cross-variable information exchange is introduced only by RASP.

4.3 Relation-Aware State Propagation

Given the variable-wise temporal embeddings $\mathbf{E} \in \mathbb{R}^{C\times D}$, we model cross-variable information through **Relation-Aware State Propagation (RASP)**. Rather than constructing interactions between all variable pairs, RASP relates each variable to a compact multivariate context and uses this relation to control how predictive information is incorporated into and propagated through the recurrent state.

For each variable, we first construct a leave-one-out context that summarizes the remaining multivariate system. A direct context is obtained by mean aggregation:

$$\mathbf{c}_i^{\text{mean}} = \frac{\sum_{j=1}^{C} \mathbf{e}_j - \mathbf{e}_i}{C-1}. \tag{8}$$

To improve the flexibility of this compact representation, we further introduce a learnable residual refinement,

$$\mathbf{c}_i = \mathbf{c}_i^{\text{mean}} + \rho\left(\frac{\sum_{j=1}^{C} \phi(\mathbf{e}_j) - \phi(\mathbf{e}_i)}{C-1}\right), \qquad \mathbf{c}_i \in \mathbb{R}^D, \tag{9}$$

where $\phi(\cdot)$ and $\rho(\cdot)$ are lightweight learnable transformations. All variable-specific contexts can be obtained from shared global aggregations without constructing a dense $C \times C$ interaction matrix.

With this compact context, RASP next determines how information from each variable should be incorporated into the recurrent propagation process. For each propagation direction $d \in \{\rightarrow, \leftarrow\}$, the relation-aware gate is computed as

$$\mathbf{g}_i^d = \sigma\Big(\mathbf{W}_e^d \mathbf{e}_i + \mathbf{W}_c^d \mathbf{c}_i + \mathbf{w}_\Delta^d \odot |\mathbf{e}_i - \mathbf{c}_i| + \mathbf{w}_\odot^d \odot (\mathbf{e}_i \odot \mathbf{c}_i) + \mathbf{b}_g^d\Big). \tag{10}$$

The current variable, its multivariate context, their absolute difference, and their element-wise interaction provide complementary relational information for controlling the state transition. In parallel, the candidate representation is generated from the current variable:

$$\mathbf{u}_i^d = \tanh\left(\mathbf{W}_u^d \mathbf{e}_i + \mathbf{b}_u^d\right). \tag{11}$$

Together, the relation-aware gate and candidate representation determine how information is retained from the propagated state and incorporated from the current variable. For the forward direction, the state update is

$$\mathbf{h}_i^{\rightarrow} = \mathbf{g}_i^{\rightarrow} \odot \mathbf{h}_{i-1}^{\rightarrow} + (\mathbf{1} - \mathbf{g}_i^{\rightarrow}) \odot \mathbf{u}_i^{\rightarrow}. \tag{12}$$

The backward direction follows the same transition over the reversed traversal, using $\mathbf{h}_{i+1}^{\leftarrow}$ as the propagated state. We initialize the boundary states as $\mathbf{h}_0^{\rightarrow} = \mathbf{0}$ and $\mathbf{h}_{C+1}^{\leftarrow} = \mathbf{0}$.

The relation-aware gate thus explicitly modulates the local state transition using variable–context relations, while the evolving recurrent state propagates information across individual variables. Importantly, both the gate and candidate representation are computed independently of the previous recurrent state. As a result, the state update takes an affine recurrent form while retaining relation-aware conditioning. This property enables the parallel evaluation introduced next.

4.4 Work-Efficient Bidirectional Propagation

Building on the affine structure of RASP, we reorganize the recurrent computation to reduce the sequential dependency depth along the variable dimension. A direct evaluation of Equation 12 processes the states sequentially and therefore incurs a dependency depth that grows linearly with the number of variables.

For either propagation direction, after indexing variables according to the corresponding traversal order, we omit the directional superscript for clarity and write the recurrent update as

$$\mathbf{h}_i = \mathbf{a}_i \odot \mathbf{h}_{i-1} + \mathbf{b}_i, \tag{13}$$

where

$$\mathbf{a}_i = \mathbf{g}_i, \qquad \mathbf{b}_i = (\mathbf{1} - \mathbf{g}_i) \odot \mathbf{u}_i. \tag{14}$$

Each variable therefore defines an element-wise affine transformation

$$\mathcal{A}_i(\mathbf{h}) = \mathbf{a}_i \odot \mathbf{h} + \mathbf{b}_i. \tag{15}$$

For two consecutive transformations, their composition is

$$(\mathbf{a}_j, \mathbf{b}_j) \circ (\mathbf{a}_i, \mathbf{b}_i) = (\mathbf{a}_j \odot \mathbf{a}_i,\ \mathbf{a}_j \odot \mathbf{b}_i + \mathbf{b}_j). \tag{16}$$

Since these affine transformations can be composed associatively, the recurrence can be evaluated using a work-efficient parallel prefix scan (Blelloch, 1990). When the number of variables is not a

power of two, we pad the variable sequence to the next power of two using the identity transform $(\mathbf{1}, \mathbf{0})$. The scan preserves $\mathcal{O}(CD)$ arithmetic work, which is linear in the number of variables, but reduces the sequential dependency path from $\mathcal{O}(C)$ to $\mathcal{O}(\log C)$ through parallel tree-structured evaluation. This shorter dependency path enables more parallel execution than direct sequential recurrence. The scan changes only the evaluation strategy and therefore preserves the same recurrent states as sequential evaluation.

The scan efficiently evaluates a fixed traversal. To reduce the directional asymmetry introduced by any single traversal, we apply RASP in both forward and backward directions. The two directions share the same multivariate context construction, while direction-specific parameters produce their corresponding gates and candidate representations.

Finally, the two directional states are fused with the original temporal representation through a residual connection:

$$\mathbf{z}_i = \mathbf{e}_i + \mathbf{W}_o \,\mathrm{Concat}\,(\mathbf{h}_i^{\rightarrow}, \mathbf{h}_i^{\leftarrow}) + \mathbf{b}_o. \tag{17}$$

Collecting all variable representations gives

$$\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_C] \in \mathbb{R}^{C \times D}, \tag{18}$$

which is subsequently mapped to the forecasting horizon by the prediction head $\mathcal{P}$.

The bidirectional traversal serves as a computational mechanism for cross-variable information propagation and does not assume that the chosen traversal reflects an intrinsic semantic ordering among variables. Additional analysis of traversal sensitivity is provided in Appendix F.

5 Experiments

We conduct extensive experiments to evaluate TimeRelay from four perspectives: (1) forecasting performance across diverse multivariate benchmarks (Section 5.2); (2) the effectiveness of RASP for cross-variable modeling (Section 5.4); (3) scalability with respect to the number of variables (Section 5.5) and computational efficiency (Appendix E); and (4) sensitivity to different variable traversals (Appendix F). We further perform ablation studies to examine the contributions of the main modules in TimeRelay (Section 5.3) and the internal design choices of RASP (Appendix D). Additional analyses investigate the sensitivity of key architectural hyperparameters (Appendix C) and the ability to exploit longer historical contexts (Appendix G).

5.1 Experimental Setup

Datasets. We evaluate TimeRelay on ten public multivariate time series forecasting benchmarks covering diverse application domains and variable dimensionalities. For long-horizon forecasting, we consider ECL, Traffic, and Solar-Energy, with prediction horizons $H \in \{96, 192, 336, 720\}$. For short-horizon forecasting, we evaluate on PEMS03, PEMS04, PEMS07, PEMS08, PEMSD7, PEMS-BAY and METR-LA, with prediction horizons $H \in \{12, 24, 48, 96\}$. We follow the established preprocessing and data splits used in previous studies for each benchmark (Liu et al., 2024). Detailed dataset statistics and preprocessing procedures are provided in Appendix B.1.

Baselines. We compare TimeRelay with fourteen representative forecasting models spanning different modeling paradigms: TimesNet (Wu et al., 2023a), iTransformer (Liu et al., 2024), TQNet (Lin et al., 2025), TimeXer (Wang et al., 2024b), ModernTCN (Luo & Wang, 2024), TimePro (Ma et al., 2025), DUET (Qiu et al., 2025), TSMixer (Ekambaram et al., 2023), SOFTS (Han et al., 2024), PatchTST (Nie et al., 2023), CycleNet (Lin et al., 2024a), TimeFilter (Hu et al., 2025), Olinear (Yue et al., 2025) and VPNet (Liu et al., 2026). The selected baselines cover attention-based, convolutional, MLP-based, variable-independent, and structured cross-variable modeling approaches, providing comparisons across different temporal and cross-variable modeling strategies.

5.2 Main Forecasting Results

Table 1 and Table 2 summarize the forecasting performance of TimeRelay and representative baselines across ten multivariate datasets using MSE and MAE. Overall, TimeRelay achieves the best

Table 1: Full results on PEMS datasets. The input length L is fixed as 96 and the results are averaged over all prediction horizons. The best results are highlighted in **bold**. 1st denotes the number of best-performing metrics.

Models	PEMS03		PEMS04		PEMS07		PEMS08		PEMSD7		1st
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	
TimesNet(2023a)	0.173	0.264	0.148	0.263	0.133	0.241	0.214	0.293	0.569	0.436	0
PatchTST(2023)	0.150	0.260	0.171	0.283	0.165	0.267	0.242	0.285	0.485	0.412	0
CycleNet(2024a)	0.118	0.225	0.119	0.231	0.113	0.189	0.150	0.228	0.503	0.429	0
SOFTS(2024)	0.104	0.210	0.102	0.208	0.087	0.184	0.138	0.219	0.503	0.429	0
TSMixer(2023)	0.158	0.263	0.129	0.247	0.117	0.221	0.201	0.294	0.488	0.394	0
OLinear(2025)	0.098	0.203	0.094	0.198	0.085	0.181	0.119	0.203	0.375	0.343	0
ModernTCN(2024)	0.111	0.226	0.103	0.217	0.118	0.208	0.230	0.268	0.374	0.370	0
VPNet(2026)	0.106	0.216	0.100	0.212	0.087	0.191	0.122	0.213	0.355	0.329	0
TimePro(2025)	0.119	0.225	0.114	0.226	0.096	0.200	0.252	0.259	0.412	0.370	0
DUET(2025)	0.110	0.219	0.113	0.227	0.091	0.198	0.130	0.229	0.360	0.336	0
TQNet(2025)	0.097	0.203	0.091	0.197	0.075	0.171	0.142	0.229	0.361	0.343	0
iTransformer(2024)	0.113	0.221	0.119	0.231	0.113	0.189	0.150	0.228	0.382	0.354	0
TimeXer(2024b)	0.111	0.223	0.110	0.227	0.093	0.199	0.233	0.269	0.337	0.319	0
TimeFilter(2025)	0.108	0.217	0.114	0.226	0.101	0.207	0.139	0.230	0.498	0.404	0
TimeRelay	**0.090**	**0.197**	**0.088**	**0.195**	**0.073**	**0.168**	**0.114**	**0.201**	**0.324**	**0.313**	**10**

Table 2: Full results on other multivariate forecasting datasets.

Models	ECL		Traffic		Solar-Energy		PEMS-BAY		METR-LA		1st
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	
TimesNet(2023a)	0.192	0.295	0.620	0.336	0.301	0.319	0.669	0.375	0.908	0.520	0
PatchTST(2023)	0.189	0.276	0.454	0.286	0.236	0.266	0.596	0.382	0.762	0.488	0
CycleNet(2024a)	0.168	0.259	0.472	0.301	0.210	0.261	0.604	0.397	0.748	0.503	0
SOFTS(2024)	0.174	0.264	0.409	**0.267**	0.229	0.256	0.604	0.397	0.748	0.503	**1**
TSMixer(2023)	0.186	0.287	0.522	0.357	0.260	0.297	0.523	0.336	0.833	0.490	0
OLinear(2025)	0.166	0.263	0.475	0.287	0.222	0.255	0.406	0.289	0.774	0.477	0
ModernTCN(2024)	0.197	0.282	0.546	0.348	0.211	0.310	0.436	0.330	0.808	0.500	0
VPNet(2026)	0.162	**0.251**	0.421	0.273	0.204	**0.207**	0.396	0.292	0.756	0.478	**2**
TimePro(2025)	0.169	0.262	0.443	0.287	0.232	0.266	0.577	0.389	0.787	0.533	0
DUET(2025)	0.170	0.265	0.469	0.289	0.220	0.263	0.447	0.326	0.813	0.487	0
TQNet(2025)	0.164	0.259	0.445	0.276	0.198	0.256	0.427	0.315	0.749	0.489	0
iTransformer(2024)	0.178	0.270	0.428	0.282	0.233	0.262	0.447	0.319	0.837	0.536	0
TimeXer(2024b)	0.171	0.270	0.466	0.287	0.202	0.269	0.394	0.288	0.750	0.496	0
TimeFilter(2025)	**0.158**	0.256	**0.407**	0.268	0.223	0.250	0.547	0.358	0.819	0.493	**2**
TimeRelay	0.167	0.261	0.427	0.281	**0.192**	0.249	**0.362**	**0.277**	**0.716**	**0.474**	**5**

performance on most datasets, demonstrating strong forecasting accuracy across datasets with substantially different numbers of variables and temporal characteristics.

Specifically, TimeRelay shows clear improvements on several representative high-dimensional benchmarks. On METR-LA, TimeRelay achieves MSE reductions of 14.5%, 5.3%, 6.0%, and 12.6% compared with iTransformer, VPNet, PatchTST, and TimeFilter, respectively. On PEMS-BAY, the corresponding reductions are 19.0%, 8.6%, 39.3%, and 33.8%. On PEMS03, TimeRelay achieves an MSE of 0.090, reducing the error by 20.4%, 15.1%, 40.0%, and 16.7% against the same four baselines. On PEMS04, TimeRelay further achieves MSE reductions of 26.1%, 12.0%, 48.5%, and 22.8%, respectively.

These results demonstrate that TimeRelay remains competitive across different forecasting paradigms, including attention-based, variable-independent, graph-based, and recent variable modeling approaches. Although TimeRelay does not achieve the best individual metrics on ECL and

Traffic, its performance remains competitive on both datasets while showing consistent improvements on the remaining benchmarks. In particular, the strong results on multiple traffic-related datasets suggest that relation-aware state-mediated propagation can effectively communicate predictive information across variables without explicitly materializing a dense pairwise interaction.

5.3 Component Ablation

We examine the contributions of the two main components in TimeRelay: multi-scale temporal enhancement and Relation-Aware State Propagation (RASP). We compare the full model with two variants that remove either the multi-scale residual enhancement or RASP, while keeping the remaining components unchanged. As shown in Table 3, removing either component consistently degrades forecasting performance. The degradation without RASP is particularly pronounced on Traffic and PEMS07, highlighting the importance of cross-variable information propagation on high-dimensional datasets. Removing the multi-scale enhancement also leads to consistent performance drops, indicating that improved variable-wise temporal representations provide complementary benefits. Overall, the results confirm the contributions of both components to TimeRelay.

Table 3: Ablation study of the main components in TimeRelay.

Variant	Solar		METR-LA		ECL		Traffic		PEMS07	
	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓
w/o Multi-Scale Enhancement	0.205	0.254	0.736	0.490	0.175	0.267	0.432	0.285	0.094	0.198
w/o RASP	0.212	0.262	0.734	0.485	0.182	0.270	0.479	0.302	0.116	0.217
TimeRelay	**0.192**	**0.249**	**0.716**	**0.474**	**0.167**	**0.261**	**0.427**	**0.281**	**0.073**	**0.168**

5.4 Analysis of Relation-Aware State Propagation

We examine whether the forecasting gains of TimeRelay mainly arise from introducing cross-variable modeling in general or from the specific relation-aware state propagation design of RASP. To isolate this effect, we keep the temporal embedding, prediction head, and all other components unchanged, and only replace the cross-variable modeling module.

We compare RASP with several representative alternatives. *None* removes cross-variable modeling; *MLP* performs feed-forward mixing across variables; *Attention* explicitly models pairwise variable interactions; *Mamba* applies state-space modeling along the variable dimension; and *BiGRU* propagates information recurrently across variables. To further isolate the effects of multivariate context and affine recurrent propagation, *Context-BiGRU* augments BiGRU with the same multivariate context used by RASP, while *PlainScan* retains the affine recurrent propagation of RASP but removes relation-aware context conditioning.

Table 4: Comparison of different cross-variable modeling mechanisms.

Cross-Variable Mixer	Solar		METR-LA		ECL		Traffic		PEMS07	
	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓
None	0.212	0.262	0.734	0.485	0.182	0.270	0.479	0.302	0.116	0.217
MLP	0.207	0.267	0.807	0.536	0.188	0.288	0.518	0.354	0.141	0.242
Attention	0.205	0.267	0.764	0.490	**0.167**	0.263	0.430	0.288	0.074	0.173
Mamba	0.214	0.272	0.742	0.475	0.174	0.267	0.442	0.281	**0.073**	0.168
BiGRU	0.199	0.257	0.756	0.481	0.169	0.265	0.436	0.282	**0.073**	0.169
Context-BiGRU	0.203	0.256	0.736	0.478	0.171	0.266	0.435	**0.279**	**0.073**	**0.168**
PlainScan	0.197	0.251	0.743	0.487	**0.167**	0.263	0.438	0.286	0.074	0.172
RASP	**0.192**	**0.249**	**0.716**	**0.474**	**0.167**	**0.261**	**0.427**	0.281	**0.073**	**0.168**

As shown in Table 4, simply introducing a cross-variable mixer does not necessarily improve forecasting performance. MLP degrades performance on several datasets compared with *None*, whereas Attention, Mamba, and BiGRU provide clear improvements on multiple datasets, confirming that the modeling mechanism matters. The controlled variants further clarify the source of the gains. Context-BiGRU improves over BiGRU in several cases, while PlainScan improves over *None* on four of the five datasets, demonstrating the benefits of multivariate context and affine state-mediated propagation. RASP further improves upon PlainScan on Solar, METR-LA, Traffic, and PEMS07,

and achieves the same MSE on ECL. It also obtains lower or equal errors than Attention, Mamba, and BiGRU across all five datasets.

Overall, the gains of RASP cannot be attributed to cross-variable mixing, recurrent or state-space modeling, or multivariate context alone. Instead, using multivariate context to modulate the affine recurrent transition provides a more effective mechanism for cross-variable information propagation.

5.5 Scaling with the Number of Variables

To assess whether RASP maintains practical computational efficiency as the variable dimension grows, we examine how different cross-variable modeling mechanisms scale with the number of variables. Using PEMS07 with 883 variables, we construct subsets with $C \in \{50, 100, 200, 400, 800\}$. To reduce the influence of hidden-dimension-dependent computation and better isolate the effect of the variable dimension, we fix the hidden dimension to $D = 64$ for all compared methods and keep the remaining computational settings identical.

We compare MLP Mixer, Attention, BiGRU, and RASP, representing feed-forward variable mixing, explicit pairwise interaction, conventional recurrent propagation, and relation-aware state-mediated propagation, respectively. For each C, we report the parameter count, runtime per epoch, and GPU memory usage under the same computational setting.

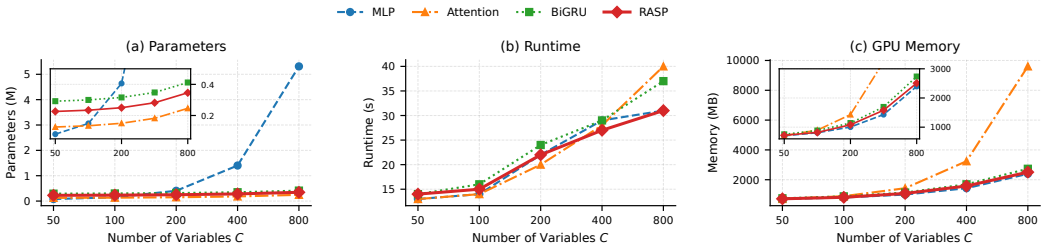


Figure 3: Computational scalability with respect to the number of variables on PEMS07.

As shown in Figure 3, the four mechanisms exhibit distinct scaling behaviors as C increases. MLP maintains low runtime and memory usage due to its simple feed-forward computation, but its parameter count grows rapidly with C, from 0.080M at $C = 50$ to 5.312M at $C = 800$. Attention shows the opposite trade-off: its parameterization remains compact because its projection parameters are largely independent of C under a fixed hidden dimension, whereas explicit pairwise interaction over a $C \times C$ space causes GPU memory to increase rapidly, reaching 9603 MB at $C = 800$.

BiGRU and RASP exhibit more moderate memory growth, while RASP also maintains competitive runtime at large C. At $C = 800$, RASP requires 0.346M parameters, 31 s of runtime, and 2509 MB of GPU memory, compared with 0.246M, 40 s, and 9603 MB for Attention, and 0.412M, 37 s, and 2735 MB for BiGRU. Relative to Attention, RASP reduces runtime by 22.5% and GPU memory by 73.9%. Although MLP remains slightly more efficient in runtime and memory, RASP requires substantially fewer parameters as C grows. Overall, RASP combines strong forecasting performance with a balanced scaling profile across parameter count, runtime, and memory, avoiding both the large memory growth of Attention and the rapidly increasing parameterization of MLP-based variable mixing.

6 Conclusion

In this work, we presented **TimeRelay**, a relation-aware recurrent framework for multivariate time series forecasting. Instead of explicitly materializing dense pairwise interactions, TimeRelay propagates predictive information across variables through the RASP module, which conditions recurrent state transitions on the relation between each variable and a compact multivariate context. The resulting affine recurrent form supports work-efficient parallel evaluation with linear arithmetic work in the number of variables and logarithmic dependency depth. Extensive experiments demonstrate strong forecasting performance together with favorable computational efficiency and scalability, while further analyses validate the effectiveness of relation-aware state-mediated propagation.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, we provide the implementation details in Appendix B and release the code at the anonymous repository: `https://anonymous.4open.science/r/TimeRelay-MTSF`.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing and manuscript organization, provide feedback on research methodology and experimental design, help interpret and summarize experimental results, and refine plotting and presentation-related code. We did not use generative AI tools to generate experimental measurements or fabricate scientific evidence. All AI-assisted content was carefully reviewed, verified, and revised by the authors. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work focuses on methodological research for multivariate time series forecasting and does not involve human subjects or the collection of personally identifiable information. Experiments are conducted on established benchmark datasets without introducing new data collection. The proposed method may be applied to real-world forecasting tasks, where prediction errors could affect downstream decisions; therefore, deployment in high-stakes settings should include appropriate domain-specific validation and safeguards.

REFERENCES

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619:1–6, 07 2023. doi: 10.1038/s41586-023-06185-3.

Guy E. Blelloch. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, 1990.

Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748 (1):96–102, 2001.

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. *ICLR*, 2024.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 459–469, 2023.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling*, 2024.

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*, 2024.

W. Daniel Hillis and Guy L. Steele, Jr. Data parallel algorithms. *Communications of the ACM*, 29 (12):1170–1183, 1986. doi: 10.1145/7902.7903.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, November 1997. ISSN 0899-7667, 1530-888X.

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. *ICML*, 2025.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, 2018.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *ICLR*, 2018.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. Segrnn: Segment recurrent neural network for long-term time series forecasting, 2023. URL https://arxiv.org/abs/2308.11200.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: enhancing time series forecasting through modeling periodic patterns. *NIPS*, 2024a.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with 1k parameters. *ICML*, 2024b.

Shengsheng Lin, Haojun Chen, Haijie Wu, Chunyun Qiu, and Weiwei Lin. Temporal query network for efficient multivariate time series forecasting. In *Forty-second International Conference on Machine Learning*, 2025.

Kun Liu, Renjun Jia, Ruifeng Yang, Xirui Zeng, Yuqi Liang, and Cen Chen. Are global dependencies necessary? scalable time series forecasting via local cross-variate modeling. In *International Conference on Learning Representations*, 2026.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: time series modeling and forecasting with sample convolution and interaction. *NeurIPS*, 2022.

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

Donghao Luo and Xue Wang. ModernTCN: A modern pure convolution structure for general time series analysis. *ICLR*, 2024. URL https://openreview.net/forum?id=vpJMJerXHU.

Xiaowen Ma, Zhenliang Ni, Shuai Xiao, and Xinghao Chen. Timepro: Efficient multivariate long-term time series forecasting with variable- and time-aware hyper-state. *ICML*, 2025.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Zheng Qian, Yan Pei, Hamidreza Zareipour, and Niya Chen. A review and discussion of decomposition-based hybrid models for wind energy forecasting applications. *Applied Energy*, 235:939–953, 02 2019. doi: 10.1016/j.apenergy.2018.10.080.

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*, pp. 1185–1196, 2025.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. *International Conference on Learning Representations*, 2023.

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. *ICLR*, 2024a.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. Timemixer++: A general time series pattern machine for universal predictive analysis. *ICLR*, 2025a.

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

Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025b.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 101–112, 2021.

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

Haixu Wu, Hang Zhou, Mingsheng Long, and Jianmin Wang. Interpretable weather forecasting for worldwide stations with a unified deep model. *Nature Machine Intelligence*, 5:1–10, 06 2023b. doi: 10.1038/s42256-023-00667-9.

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*, pp. 753–763, 2020.

Xueyan Yin, Genze Wu, Jinze Wei, Yanming Shen, Heng Qi, and Baocai Yin. Deep learning on traffic prediction: Methods, analysis, and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 23:4927–4943, 06 2022. doi: 10.1109/TITS.2021.3054840.

Wenzhen Yue, Yong Liu, Hao Wang, Haoxuan Li, Xianghua Ying, Ruohao Guo, Bowei Xing, and Ji Shi. Olinear: A linear model for time series forecasting in orthogonally transformed domain. *Advances in Neural Information Processing Systems*, 38:60573–60632, 2025.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *AAAI*, 2023.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. *ICLR*, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecastings. In *International Conference on Machine Learning*, 2022.

A Related Work

A.1 Multivariate Time Series Forecasting

Multivariate time series forecasting plays an important role in a wide range of applications, including traffic prediction (Yin et al., 2022), energy forecasting (Qian et al., 2019), and weather forecasting (Wu et al., 2023b; Bi et al., 2023). Recent studies have explored diverse architectures to capture temporal patterns and dependencies among multiple variables.

A broad class of variable-dependent methods jointly processes multivariate observations while primarily focusing on temporal representation learning. Representative approaches include Transformer-based models such as Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021), and FEDformer (Zhou et al., 2022), as well as architectures with specialized temporal modeling mechanisms, including MICN (Wang et al., 2023), TimesNet (Wu et al., 2023a), Pathformer (Chen et al., 2024), TimeMixer (Wang et al., 2024a), and TimeMixer++ (Wang et al., 2025a). These methods develop diverse mechanisms for capturing temporal patterns, including decomposition, frequency modeling, convolution, two-dimensional temporal variation, and multi-scale representation learning. Their primary architectural emphasis is generally placed on temporal dependency modeling, rather than explicitly designing mechanisms for variable-to-variable information exchange.

In contrast, variable-independent methods model each variable separately without explicitly exchanging information across variables. Representative approaches include DLinear (Zeng et al., 2023), PatchTST (Nie et al., 2023), SegRNN (Lin et al., 2023), SparseTSF (Lin et al., 2024b), and CycleNet (Lin et al., 2024a). Such designs avoid the cost of cross-variable interaction and can provide strong forecasting performance, but they do not explicitly exploit information shared among different variables.

Another major line of research explicitly models dependencies or exchanges information across variables. Attention-based methods such as iTransformer (Liu et al., 2024), Crossformer (Zhang & Yan, 2023), TQNet (Lin et al., 2025), and TimeXer (Wang et al., 2024b) construct interactions between variable representations through attention mechanisms. MLP-based approaches such as TSMixer (Ekambaram et al., 2023) and SOFTS (Han et al., 2024) perform cross-variable mixing using MLP transformations, while ModernTCN (Luo & Wang, 2024) captures variable interactions through convolutional operations. State-space-based approaches such as TimePro (Ma et al., 2025) and S-Mamba (Wang et al., 2025b) further explore state-space mechanisms for multivariate modeling. Graph-based methods, including MTGNN (Wu et al., 2020) and TimeFilter (Hu et al., 2025), represent cross-variable dependencies through structured graph interactions, while DUET (Qiu et al., 2025) introduces clustering-based mechanisms to organize variable relationships.

These approaches demonstrate the importance of cross-variable information, but many representative methods identify or propagate such information through explicit interactions between variable representations. When a complete pairwise interaction structure is materialized, the candidate interaction space grows quadratically with the number of variables. TimeRelay instead propagates predictive information through recurrent states whose transitions are conditioned on a compact multivariate context, avoiding the need to explicitly construct a dense variable-pair interaction matrix.

A.2 Recurrent and State-Space Modeling

Recurrent neural networks provide a natural mechanism for transmitting information through an evolving hidden state. Classical recurrent architectures such as LSTM (Hochreiter & Schmidhuber, 1997) and GRU (Cho et al., 2014) update a compact state sequentially and have been widely used for sequence modeling and time series forecasting. Recurrent forecasting models such as DeepAR (Salinas et al., 2020) and SegRNN (Lin et al., 2023) further demonstrate the effectiveness of recurrent state propagation for temporal prediction.

More recently, state-space models have emerged as an efficient approach to sequence modeling. Mamba (Gu & Dao, 2024), for example, introduces input-dependent selective state-space dynamics. State-space mechanisms have also been adapted to multivariate time series forecasting. S-Mamba (Wang et al., 2025b) employs bidirectional state-space modeling to capture inter-variable dependencies, while TimePro (Ma et al., 2025) further explores variable- and time-aware state modeling for multivariate forecasting.

TimeRelay differs from these approaches primarily in how cross-variable state transitions are conditioned and structured. Its Relation-Aware State Propagation (RASP) explicitly conditions each recurrent transition on the relation between the current variable representation and a compact multivariate context. The resulting recurrence retains an affine form whose transformations can be composed associatively, enabling work-efficient parallel evaluation. In this way, the variable–context relation directly controls how information is incorporated into and propagated through the recurrent state, without requiring a dense variable-pair interaction matrix.

B Implementation Details

B.1 Dataset Details

The details of the experimental datasets are summarized as follows: (1) *ECL* (Wu et al., 2021): Records hourly electricity consumption of 321 customers from 2012 to 2014. (2) *Traffic* (Wu et al., 2021): Hourly road occupancy rates from sensors on San Francisco Bay area freeways, collected by the California Department of Transportation. (3) *Solar-Energy* (Lai et al., 2018): Solar power production of 137 PV plants in 2006, sampled every 10 minutes. (4) *PEMS* (Liu et al., 2022): Public traffic network data from California, collected in 5-minute intervals, including subsets PEMS03, PEMS04, PEMS07, and PEMS08. (5) *PeMSD7(M)* (Chen et al., 2001): Data from the Caltrans PeMS system, containing 228 time-series collected at 5-minute intervals. (6) *PEMS-BAY* (Li et al., 2018): Traffic data from 325 sensors in the Bay Area, collected over 6 months (Jan 1, 2017, to May 31, 2017) by CalTrans Performance Measurement System (PeMS). (7) *METR-LA* (Li et al., 2018): Traffic data from 207 sensors on Los Angeles County highways, collected over 4 months (Mar 1, 2012, to Jun 30, 2012).

We adopt the same data processing and train-validation-test split protocol from iTransformer (Liu et al., 2024). For long-term forecasting, the lookback length is fixed at 96 for the ECL, Traffic, and Solar-Energy datasets, with prediction lengths of $\{96, 192, 336, 720\}$. For short-term forecasting, the lookback length is also fixed at 96 for the PEMS03, PEMS04, PEMS07, PEMS08, PeMSD7(M), PEMS-BAY, and METR-LA datasets, with prediction lengths of $\{12, 24, 48, 96\}$. Detailed dataset statistics are provided in Table 5.

Table 5: Detailed dataset descriptions. *Dim* denotes the variable number of each dataset. *Dataset Size* denotes the total number of time points in (Train, Validation, Test) split respectively. *Prediction Length* denotes the prediction length in each dataset. *Frequency* denotes the sampling interval of time points.

Dataset	Dim	Prediction Length	Dataset Size	Frequency	Information
ECL	321	{96, 192, 336, 720}	(18317, 2633, 5261)	Hourly	Electricity
Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	Hourly	Transportation
Solar-Energy	137	{96, 192, 336, 720}	(36601, 5161, 10417)	10min	Energy
PEMS03	358	{12, 24, 48, 96}	(15617, 5135, 5135)	5min	Transportation
PEMS04	307	{12, 24, 48, 96}	(10172, 3375, 3375)	5min	Transportation
PEMS07	883	{12, 24, 48, 96}	(16911, 5622, 5622)	5min	Transportation
PEMS08	170	{12, 24, 48, 96}	(10690, 3548, 3548)	5min	Transportation
PeMSD7(M)	228	{12, 24, 48, 96}	(8763, 1256, 2523)	5min	Transportation
PEMS-BAY	325	{12, 24, 48, 96}	(36374, 5201, 10412)	5min	Transportation
METR-LA	207	{12, 24, 48, 96}	(23883, 3417, 6843)	5min	Transportation

B.2 Implementation Details

All experiments are implemented in PyTorch (Paszke et al., 2019) and conducted on an NVIDIA H20 GPU. The batch size is uniformly set to 32. Models are trained for at most 15 epochs, with early stopping if the validation loss does not improve for three consecutive epochs. We use the mean

squared error (MSE) loss for training and the Adam optimizer (Kingma & Ba, 2015) with an initial learning rate of 0.001. Forecasting performance is evaluated using Mean Squared Error (MSE) and Mean Absolute Error (MAE). Whenever published results are available under the same data preprocessing, input length, prediction horizon, and evaluation protocol, we report the corresponding results from the original papers. Otherwise, we reproduce the methods using their official implementations under the same experimental setting as TimeRelay, with hyperparameters selected on the validation set. For TimeRelay, the number of model layers is selected from $\{1, 2, 3, 4\}$ based on the validation set. The hidden dimension is fixed to 512 for all datasets, and the multi-scale temporal embedding uses a unified set of scales $\{1, 4, 12, 24\}$. The dropout rate is uniformly set to 0.1.

C Parameter Sensitivity

We further investigate the sensitivity of TimeRelay to three key architectural choices: the hidden dimension D, the number of model layers, and the multi-scale configuration P. We conduct the analysis on four representative datasets, ECL, PEMS-BAY, METR-LA, and PEMS04. For each setting, all other configurations are kept unchanged, and performance is averaged over all prediction horizons.

Hidden Dimension. We vary the hidden dimension over $D \in \{64, 128, 256, 512, 1024\}$. As shown in Figure 4, increasing D from 64 to moderate values generally improves forecasting performance on ECL, PEMS-BAY, and PEMS04, suggesting that additional representation capacity is beneficial for capturing temporal and cross-variable patterns, while the performance on METR-LA remains relatively stable. Among the evaluated settings, $D = 512$ provides consistently strong results, achieving the best MSE on all four datasets and the best or near-best MAE. Further increasing the hidden dimension to 1024 does not provide additional benefits and leads to noticeable degradation on several datasets, particularly PEMS-BAY and PEMS04. This indicates that increasing model capacity beyond a moderate scale is not necessarily beneficial for TimeRelay. Based on this overall performance trade-off, we set $D = 512$ as the default hidden dimension in the main experiments.

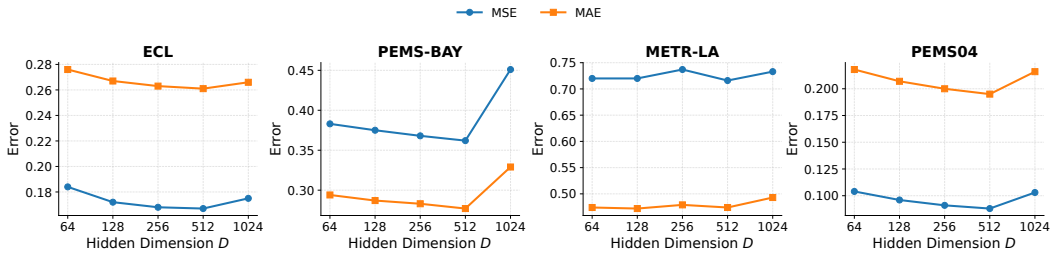


Figure 4: Sensitivity of TimeRelay to the hidden dimension D.

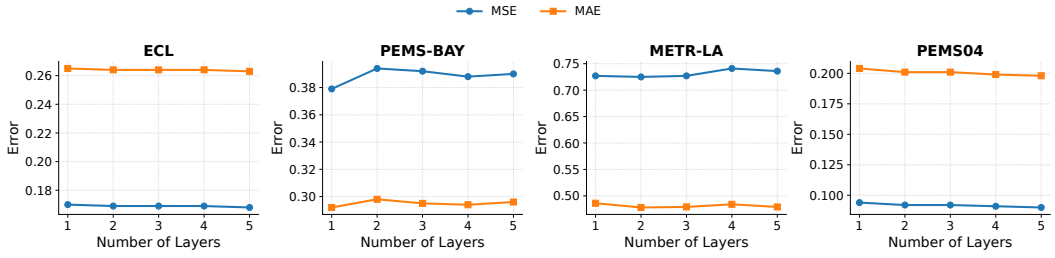


Figure 5: Sensitivity of TimeRelay to the number of model layers.

Number of Layers. We further vary the number of model layers from 1 to 5. As shown in Figure 5, the forecasting performance remains relatively stable across different depths, with only small variations on most datasets. No single depth is uniformly optimal: shallower models perform well on PEMS-BAY and METR-LA, while deeper configurations provide slightly better results on ECL

and PEMS04. Increasing the model depth allows cross-variable information to be progressively refined through multiple RASP layers, but further stacking does not consistently improve forecasting performance across datasets. Overall, these results indicate that TimeRelay is not highly sensitive to the number of layers and maintains stable performance over a relatively broad range of model depths.

Multi-Scale Configuration. We further examine the effect of temporal scale selection by comparing individual scales $p \in \{1, 4, 12, 24\}$ with the default multi-scale configuration $P = \{1, 4, 12, 24\}$. As shown in Figure 6, $p = 12$ generally performs best among the individual scales, indicating that an intermediate temporal granularity is particularly effective across the evaluated datasets. Combining all four scales further improves the overall performance, achieving the best or tied-best results across all reported metrics. This suggests that different temporal resolutions provide complementary information for variable-wise temporal representation. Accordingly, we use $P = \{1, 4, 12, 24\}$ as the default configuration in the main experiments.

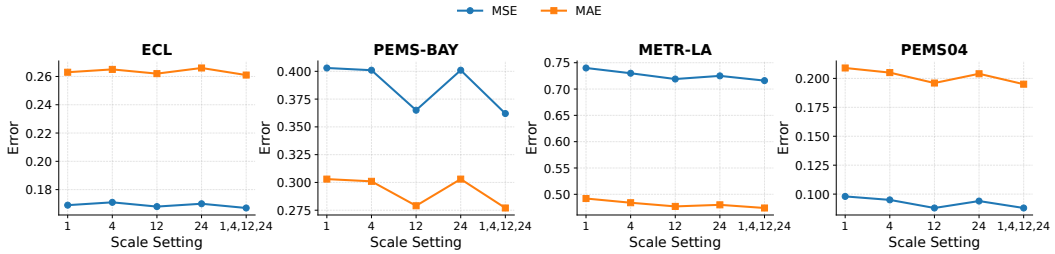


Figure 6: Sensitivity of TimeRelay to different temporal scale configurations in the residual multi-scale embedding.

D More Analysis of RASP

D.1 Effect of Multivariate Context

We further examine how the multivariate context in RASP should be constructed. In particular, we compare recurrent propagation without contextual conditioning against a simple mean context and the proposed learnable context refinement.

The *No Context* variant removes contextual conditioning from the state transition, while *Mean Context* uses the leave-one-out mean of the remaining variables. *Learnable Context* further augments this mean representation with the residual refinement introduced in Section 4.3.

Table 6: Effect of different multivariate context construction strategies in RASP.

Context Strategy	Solar		METR-LA		ECL		Traffic		PEMS07	
	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓
No Context	0.197	0.251	0.743	0.487	**0.167**	0.263	0.438	0.286	0.074	0.172
Mean Context	0.196	0.252	0.734	0.485	**0.167**	0.264	0.428	0.282	**0.072**	**0.168**
Learnable Context	**0.192**	**0.249**	**0.716**	**0.474**	**0.167**	**0.261**	**0.427**	**0.281**	0.073	**0.168**

As shown in Table 6, the simple mean context already improves performance on METR-LA, Traffic, and PEMS07, but its effect is less consistent on Solar and ECL. The proposed learnable refinement provides a more stable improvement: compared with *Mean Context*, it improves both MSE and MAE on Solar, METR-LA, and Traffic, achieves the same MSE with a lower MAE on ECL, and remains competitive on PEMS07. Overall, it improves seven of the ten reported metrics over mean aggregation, matches two, and is slightly worse on only one. These results show that the construction of the multivariate context matters. A fixed mean summary can provide useful contextual information, whereas the learnable residual refinement enables the context to better adapt to the forecasting objective and provides a more effective conditioning signal for RASP.

D.2 Effect of Relation Terms

We further investigate how different variable–context relation terms contribute to the relation-aware transition in RASP. Starting from the current variable representation, we progressively introduce the multivariate context, absolute difference, and element-wise interaction, while keeping the recurrent propagation mechanism and all other model components unchanged.

Table 7: Ablation study of the relation terms used in the RASP gate.

Relation Terms	Solar		METR-LA		ECL		Traffic		PEMS07	
	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓
$\{\mathbf{e}_i\}$	0.197	0.251	0.743	0.487	0.167	0.263	0.438	0.286	0.074	0.172
$\{\mathbf{e}_i, \mathbf{c}_i\}$	0.196	0.251	0.723	0.485	**0.166**	**0.261**	0.428	**0.281**	**0.072**	**0.168**
$\{\mathbf{e}_i, \mathbf{c}_i, \lvert\mathbf{e}_i - \mathbf{c}_i\rvert\}$	0.198	0.251	0.723	0.475	0.167	0.262	0.428	**0.281**	0.073	**0.168**
$\{\mathbf{e}_i, \mathbf{c}_i, \lvert\mathbf{e}_i - \mathbf{c}_i\rvert, \mathbf{e}_i \odot \mathbf{c}_i\}$	**0.192**	**0.249**	**0.716**	**0.474**	0.167	**0.261**	**0.427**	**0.281**	0.073	**0.168**

As shown in Table 7, introducing the multivariate context $\mathbf{c}_i$ into the gate already improves over using $\mathbf{e}_i$ alone on most datasets, with clear gains on METR-LA, Traffic, and PEMS07. This indicates that conditioning the gate jointly on the current variable and its multivariate context is an important component of RASP. The additional relation terms provide complementary rather than uniformly monotonic gains. Adding the absolute difference $|\mathbf{e}_i - \mathbf{c}_i|$ notably improves MAE on METR-LA, while producing limited changes on several other datasets. Further incorporating the element-wise interaction $\mathbf{e}_i \odot \mathbf{c}_i$ improves or matches the three-term variant across all reported metrics. The full set of relation terms consequently achieves the best or tied-best performance on eight of the ten metrics.

These results suggest that different variable–context relation terms provide complementary information for computing the RASP gate. The context term provides the primary gain, while the difference and multiplicative terms provide additional cues about discrepancy and feature-wise interaction. Although their contributions vary across datasets, combining them allows the gate to more effectively regulate recurrent state propagation.

D.3 Transferability of RASP

To examine whether RASP is specific to the TimeRelay architecture, we further integrate it into two representative variable-independent forecasting models, CycleNet and PatchTST, while keeping their original temporal modeling components unchanged.

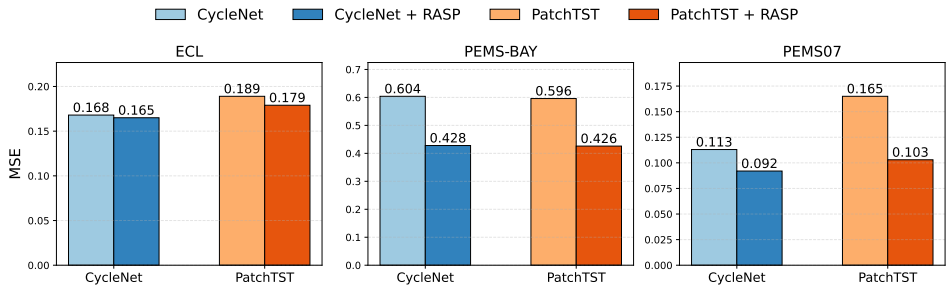


Figure 7: Transferability of RASP to variable-independent forecasting backbones.

As shown in Figure 7, incorporating RASP consistently improves both backbones across all three datasets. For CycleNet, the average MSE decreases from 0.168 to 0.165 on ECL, from 0.604 to 0.428 on PEMS-BAY, and from 0.113 to 0.092 on PEMS07, corresponding to relative reductions of 1.8%, 29.1%, and 18.6%, respectively. Similar improvements are observed for PatchTST, where RASP reduces the MSE from 0.189 to 0.179 on ECL, from 0.596 to 0.426 on PEMS-BAY, and from 0.165 to 0.103 on PEMS07, yielding reductions of 5.3%, 28.5%, and 37.6%, respectively.

These results provide additional evidence that RASP can complement variable-independent temporal modeling and is not limited to the specific backbone used in TimeRelay.

E COMPUTATIONAL EFFICIENCY ANALYSIS

A central motivation of TimeRelay is to propagate predictive information across variables without explicitly materializing dense pairwise interactions. The scalability of different cross-variable modeling mechanisms with respect to the number of variables is examined in Section 5.5. Here, we provide complementary computational analyses from two perspectives: the practical efficiency of different cross-variable modeling mechanisms and the efficiency of alternative implementations of the recurrent propagation. Together, these experiments further characterize the computational behavior of RASP and the benefit of its work-efficient parallel evaluation.

For all experiments, we fix the hidden dimension to $D = 64$ for all compared methods. This controlled setting reduces the influence of hidden-dimension-dependent computation and allows the computational differences among cross-variable modeling mechanisms to be examined directly.

Efficiency of Cross-Variable Modeling. We first replace RASP with representative cross-variable modeling mechanisms while keeping the remaining architecture unchanged. Specifically, we compare MLP, Attention, bidirectional GRU (BiGRU), and RASP, representing feed-forward variable mixing, explicit pairwise interaction, conventional recurrent propagation, and relation-aware state-mediated propagation, respectively. This controlled comparison examines their computational trade-offs under the same temporal embedding and prediction head.

We report the number of parameters, runtime per epoch, and GPU memory usage on ECL, Traffic, and PEMS07, which contain 321, 862, and 883 variables, respectively. These datasets provide representative high-dimensional settings for comparing the practical costs of different cross-variable modeling mechanisms.

Table 8: Computational efficiency of different cross-variable modeling mechanisms.

Method	ECL			Traffic			PEMS07		
	Params. (M)	Time (s) ↓	Mem. (MB) ↓	Params. (M)	Time (s) ↓	Mem. (MB) ↓	Params. (M)	Time (s) ↓	Mem. (MB) ↓
MLP	0.932	30	1291	6.147	30	2547	6.444	32	2599
Attention	0.170	32	2417	0.256	33	10705	0.259	41	11427
BiGRU	0.336	31	1513	0.422	32	2931	0.426	39	2971
RASP	0.269	31	1381	0.356	30	2841	0.359	33	2727

As shown in Table 8, the compared mechanisms exhibit distinct computational trade-offs. MLP achieves low runtime and memory consumption, but its parameter count increases substantially on the higher-dimensional Traffic and PEMS07 datasets, reaching more than 6M parameters. Attention maintains the smallest parameter count, but its explicit pairwise interaction incurs substantially higher GPU memory usage, exceeding 10 GB on Traffic and PEMS07. BiGRU avoids this dense pairwise memory cost but requires higher runtime on PEMS07 due to sequential recurrent evaluation. RASP provides a favorable balance among these computational dimensions. Its parameter count remains close to those of Attention and BiGRU, while its memory usage is substantially lower than Attention on Traffic and PEMS07. On PEMS07, RASP reduces runtime from 41 to 33 seconds compared with Attention and from 39 to 33 seconds compared with BiGRU. Although MLP remains slightly faster in some settings, it requires considerably more parameters on high-dimensional datasets. Overall, RASP avoids the large memory cost of explicit pairwise interaction while retaining competitive runtime and compact parameterization.

Efficiency of Affine Scan Evaluation. The previous comparison considers the overall cost of different cross-variable modeling mechanisms. We next isolate the evaluation strategy used by RASP. Recall that its recurrent transition can be written as

$$\mathbf{h}_i = \mathbf{a}_i \odot \mathbf{h}_{i-1} + \mathbf{b}_i, \tag{19}$$

where $\mathbf{a}_i = \mathbf{g}_i$ and $\mathbf{b}_i = (1 - \mathbf{g}_i) \odot \mathbf{u}_i$. Because the corresponding affine transformations admit associative composition, the same recurrence can be evaluated using different scan strategies.

We compare three implementations. Sequential directly evaluates the recurrence along the variable dimension, requiring $\mathcal{O}(C)$ arithmetic work and $\mathcal{O}(C)$ dependency depth. Hillis–Steele (Hillis & Steele, 1986) applies a parallel prefix scan, reducing the ideal dependency depth to $\mathcal{O}(\log C)$

at the cost of $\mathcal{O}(C \log C)$ arithmetic work. Work-Efficient uses a Blelloch-style tree-based up-sweep/down-sweep scan (Blelloch, 1990), retaining $\mathcal{O}(\log C)$ ideal dependency depth while requiring only $\mathcal{O}(C)$ arithmetic work.

Table 9: Efficiency of different evaluation strategies for the same relation-aware affine recurrence.

Strategy	ECL		Traffic		PEMS07	
	Time (s) ↓	Mem. (MB) ↓	Time (s) ↓	Mem. (MB) ↓	Time (s) ↓	Mem. (MB) ↓
Sequential	162	1399	269	2887	350	2771
Hillis–Steele	31	1739	31	3753	34	3795
Work-Efficient	31	1381	30	2841	33	2727

Table 9 demonstrates the importance of parallel evaluation for the affine recurrence. Sequential evaluation becomes increasingly expensive on datasets with more variables, requiring 162 s on ECL and 350 s on PEMS07. Both parallel scan strategies substantially reduce the runtime by replacing the linear dependency chain with logarithmic ideal dependency depth. Hillis–Steele achieves runtime comparable to the work-efficient scan but requires consistently higher GPU memory in our implementation, reaching 3795 MB on PEMS07. In contrast, the work-efficient implementation achieves the lowest or tied-lowest runtime on all three datasets while also using the least peak memory among the three strategies. On PEMS07, for example, it reduces runtime from 350 s under sequential evaluation to 33 s, while using 2727 MB of GPU memory.

Since all three strategies compute the same affine recurrence, we do not compare forecasting accuracy in Table 9. Their outputs are numerically equivalent up to floating-point differences. These results therefore isolate the computational benefit of work-efficient parallel evaluation without changing the underlying RASP transition.

F Sensitivity to Variable Traversal

RASP propagates cross-variable information through recurrent states along a chosen computational traversal of the variable dimension. Unlike temporal observations, however, variables in multivariate time series generally do not possess a canonical sequential ordering. It is therefore important to examine whether the effectiveness of TimeRelay depends strongly on a particular variable traversal. We evaluate the model under several alternative traversal orders to characterize its empirical robustness to this choice. We emphasize that this analysis studies *traversal sensitivity*, rather than claiming permutation invariance.

F.1 Different Variable Traversals

We investigate whether the performance of TimeRelay depends strongly on a particular variable traversal. Specifically, we compare the original order, the reversed order, a random permutation, and a correlation-based ordering. For each setting, the corresponding variable order is fixed throughout training, validation, and testing. The random traversal uses a random permutation, while the correlation-based traversal is constructed using correlations estimated only from the training split. All other model configurations and training settings remain unchanged.

Table 10: Sensitivity of TimeRelay to different variable traversals.

Variable Traversal	Solar		METR-LA		ECL		Traffic		PEMS07	
	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓
Original	0.192	0.249	0.716	0.474	0.167	0.261	0.427	0.281	0.073	0.168
Reverse	0.195	0.249	0.726	0.474	0.166	0.261	0.427	0.281	0.074	0.169
Random	0.197	0.250	0.719	0.478	0.166	0.260	0.427	0.281	0.074	0.169
Correlation-based	0.195	0.246	0.722	0.470	0.166	0.261	0.442	0.284	0.078	0.173

As shown in Table 10, TimeRelay maintains comparable performance under different variable traversals, although no single ordering is uniformly optimal across all datasets and metrics. The original traversal achieves the lowest MSE on Solar and METR-LA, and obtains the best or tied-best results

on both metrics of Traffic and PEMS07. Meanwhile, the correlation-based ordering achieves the lowest MAE on Solar and METR-LA, while the random traversal obtains the lowest MAE on ECL. On ECL, the MSE varies by only 0.001 across all four traversals, and the results on Traffic are also nearly identical among the original, reverse, and random orders. In contrast, the correlation-based traversal slightly degrades performance on Traffic and PEMS07, indicating that arranging adjacent variables according to pairwise correlation does not necessarily lead to better forecasting performance.

Overall, these results suggest that TimeRelay does not rely on a specially optimized variable ordering to propagate useful cross-variable information. Different traversals can lead to small performance variations, but the model remains effective under the original, reversed, and randomly permuted orders. This robustness is consistent with the bidirectional state-mediated design, which reduces dependence on a particular propagation orientation, while not implying permutation invariance.

F.2 Effect of Bidirectional Propagation

We further examine whether propagating information in both traversal directions helps reduce the directional asymmetry introduced by a single variable ordering. Specifically, we compare forward-only, backward-only, and bidirectional variants of RASP while keeping all other model components and training settings unchanged.

Table 11: Effect of propagation direction in RASP.

Propagation Direction	Solar		METR-LA		ECL		Traffic		PEMS07	
	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓
Forward Only	0.195	0.253	0.716	0.472	0.167	0.263	0.436	0.289	0.077	0.174
Backward Only	0.198	0.253	0.718	0.475	0.167	0.262	0.439	0.290	0.074	0.171
Bidirectional	0.192	0.249	0.716	0.474	0.167	0.261	0.427	0.281	0.073	0.168

As shown in Table 11, bidirectional propagation provides the most consistent overall performance across the five datasets. It achieves the best MSE and MAE on Solar, Traffic, and PEMS07, the best MAE on ECL, and ties with the forward-only variant for the best MSE on METR-LA. On ECL, all three variants obtain the same MSE, while the bidirectional variant further reduces MAE. The improvements are particularly clear on Traffic and PEMS07, where using both directions consistently outperforms either single-direction alternative on both metrics. Nevertheless, the forward-only variant achieves a slightly lower MAE on METR-LA.

Overall, these results suggest that bidirectional propagation reduces the model's dependence on a particular traversal orientation and provides a more consistent mechanism for cross-variable information propagation. Its benefit is therefore reflected primarily in improved robustness across datasets and metrics, rather than uniform superiority on every individual metric.

G Longer input length

We further investigate whether TimeRelay can effectively utilize longer historical contexts by varying the input length over $L \in \{96, 192, 336, 576, 720\}$. A longer input window provides access to a larger amount of historical information, and a model that can effectively exploit long-range context is therefore expected to benefit from such additional observations. As shown in Figure 8, increasing the input length generally improves the forecasting performance of TimeRelay on both ECL and Traffic. The performance gains become smaller for longer input lengths, suggesting that the benefit of additional historical context gradually saturates. Compared with representative baselines, TimeRelay exhibits a more stable improvement as the historical window increases and achieves the best or tied-best MSE across all evaluated input lengths on both datasets. These results demonstrate that TimeRelay can effectively utilize extended historical information while maintaining robust forecasting performance across different input lengths.

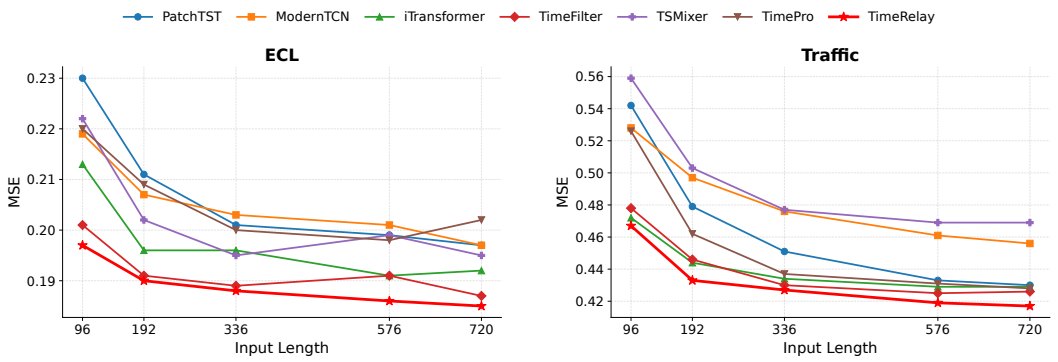
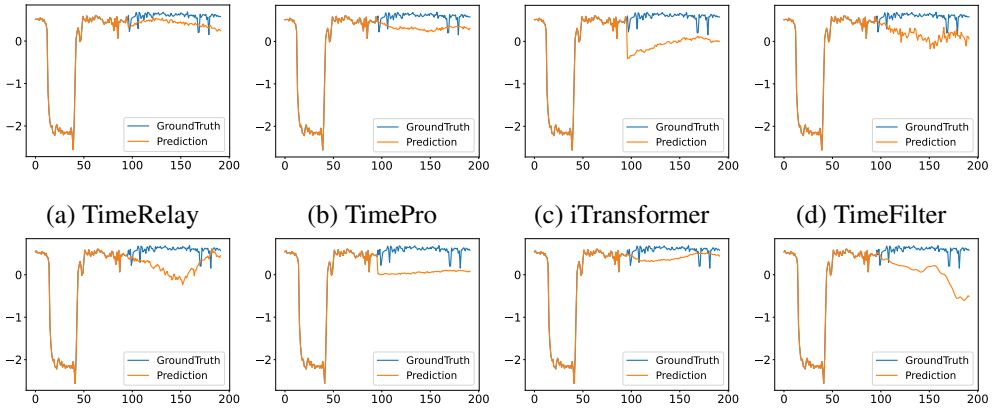


Figure 8: Forecasting performance under different input lengths on ECL and Traffic.

H SHOWCASES

To provide a qualitative comparison of forecasting performance, we present representative prediction showcases from the test sets. We compare TimeRelay with several strong baselines, including CycleNet, TimePro, TimeFilter, iTransformer, ModernTCN, TimesNet, and PatchTST, across multiple datasets. These methods cover diverse forecasting paradigms, including attention-based, graph-based, convolution-based, Mamba-based, variable-dependent, and variable-independent modeling, providing a broad comparison with different architectural designs.

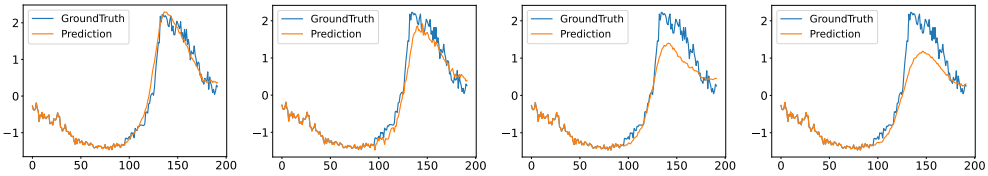
As illustrated in Figure 9, Figure 10 and Figure 11, TimeRelay generally follows the ground-truth trajectories more closely and captures both overall temporal trends and local variations across the selected examples. In particular, its predictions exhibit smaller deviations from the ground truth around several rapidly changing regions, while preserving the overall temporal evolution of the series. These qualitative results further complement the quantitative comparisons and illustrate the effectiveness of TimeRelay in forecasting diverse multivariate time series.



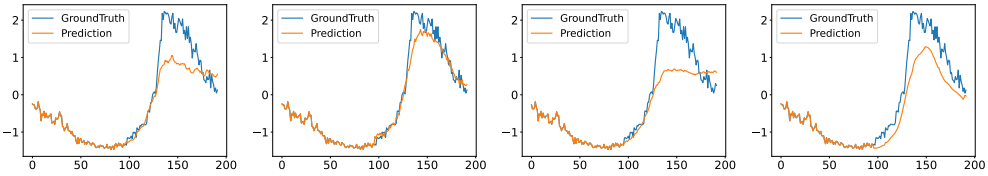
(a) TimeRelay (b) TimePro (c) iTransformer (d) TimeFilter

(e) CycleNet (f) ModernTCN (g) PatchTST (h) TimesNet

Figure 9: Showcases on METR-LA dataset. The lookback length is 96 and prediction length is 96.

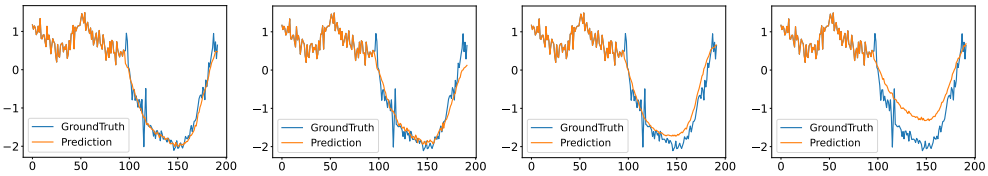


(a) TimeRelay (b) TimePro (c) iTransformer (d) TimeFilter

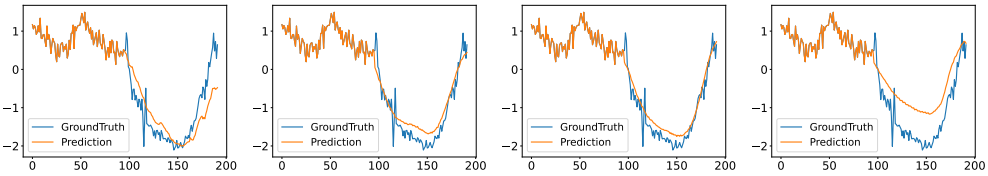


(e) CycleNet (f) ModernTCN (g) PatchTST (h) TimesNet

Figure 10: Showcases on PEMS03 dataset. The lookback length is 96 and prediction length is 96.



(a) TimeRelay (b) TimePro (c) iTransformer (d) TimeFilter



(e) CycleNet (f) ModernTCN (g) PatchTST (h) TimesNet

Figure 11: Showcases on PEMS07 dataset. The lookback length is 96 and prediction length is 96.