# BRIDGING REGULAR AND IRREGULAR TIME SERIES: A UNIFIED PERSPECTIVE

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Most time-series forecasting models assume that all variables are observed on a shared grid with uniform temporal spacing. Real-world systems often violate this assumption: in irregular multivariate time series (IMTS), variables are observed asynchronously at non-uniform timestamps, with observation frequencies varying across variables. Canonical pre-alignment (CPA) maps these observations to a shared time–variable grid, thereby placing regular and irregular time series in a common representation. This representation raises a fundamental question: *are forecasting models developed for regular time series truly incapable of handling irregular observations?* To address this question, we establish a unified perspective on representative models developed for irregular time series by organizing them according to a common four-stage pipeline: normalization, input embedding, dependency modeling, and forecast generation. By combining this perspective with controlled experiments, we identify the key factors for effective irregular forecasting and develop BridgeTST from these findings. Across five irregular and five regular datasets, BridgeTST achieves strong forecasting performance in both settings. In contrast, representative irregular forecasting models generally perform poorly in the regular setting. Building on these findings, we further examine the boundary between regular and irregular forecasting.

## 1 INTRODUCTION

Time series forecasting supports decision-making in domains such as energy management, transportation, healthcare, and climate science. Forecasting methods have evolved from classical statistical models (Box et al., 2015; Hamilton, 2020) to deep-learning architectures (Zeng et al., 2023; Wang et al., 2024; Nie et al., 2023; Liu et al., 2024) and, more recently, time-series foundation models (Liu et al., 2025b; Ansari et al., 2025; Liu et al., 2026b). Despite this progress, most models still assume that all variables are observed on a shared temporal grid with uniform spacing.

Many real-world systems produce irregular multivariate time series (IMTS), in which variables are observed at non-uniform timestamps and observation frequencies vary across variables. Existing IMTS forecasting methods generally follow one of two strategies. Some methods model asynchronous observations directly through recurrent dynamics, set-based formulations, or graph structures (Horn et al., 2020; Che et al., 2018; Kidger et al., 2020; Yalavarthi et al., 2024; Li et al., 2025; Zhang et al., 2022). These methods avoid pre-alignment but must process observations in their original asynchronous form, which incurs substantial computational costs for long inputs with high observation density. We refer to this strategy as direct asynchronous (non-CPA-based) modeling. An alternative strategy first applies canonical pre-alignment (CPA) to map asynchronous observations onto a shared time–variable grid before forecasting (Zhang et al., 2024; Liu et al., 2025a; Zhang et al., 2023; Qiu et al., 2026; Zhou et al., 2026), as illustrated in Figure 1.

The resulting grid provides a shared time–variable grid for both regular and irregular time series. Sampling intervals reflect temporal regularity, while the observation mask records which entries are observed; missing entries can occur in either setting, and missing ratios may differ substantially across variables. Thus, CPA gives both settings the same input organization while allowing their sampling intervals and observation patterns to differ. This view raises a fundamental question:



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.

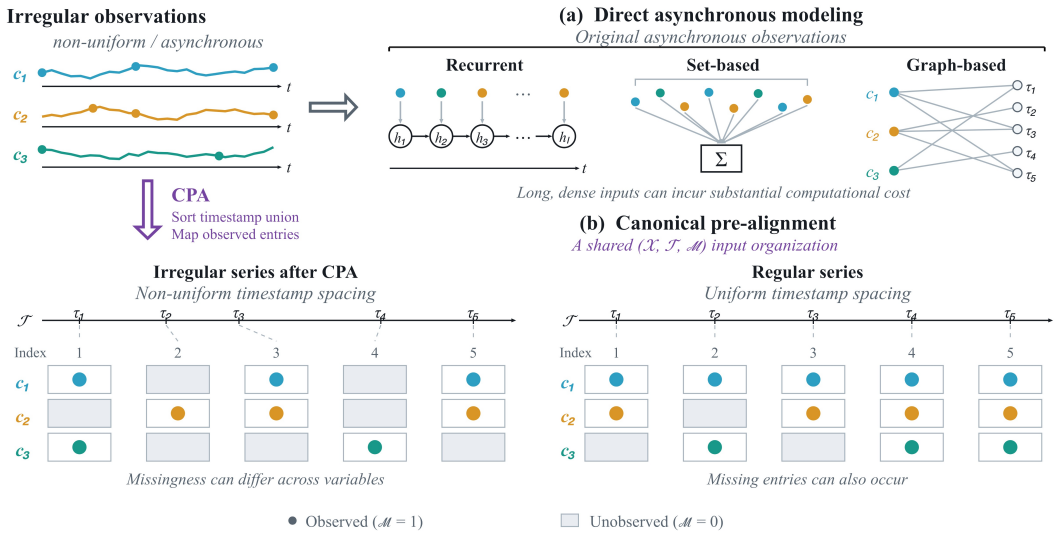


Figure 1: Two strategies for modeling irregular observations: direct asynchronous modeling and canonical pre-alignment (CPA). Direct models process observations at their original, asynchronous timestamps using recurrent, set-based, or graph-based structures. CPA maps the observations to a shared time–variable grid.

*Are regular models truly incapable of handling irregular observations?*

Throughout the paper, we use irregular models to mean models designed for irregular time series and regular models to mean models designed for regular time series. To address this question, we introduce a unified perspective on representative irregular models. Although these models use substantially different architectures, we organize their computations around a common four-stage pipeline—normalization, input embedding, dependency modeling, and forecast generation—and use this pipeline to analyze the design principles shared by these models. This perspective provides the research community with a clearer understanding of the design principles underlying irregular forecasting and guides our subsequent analysis and experiments.

Building on this perspective, we conduct controlled experiments to identify the factors that matter most for irregular forecasting. These experiments show that a general-purpose forecasting pipeline can effectively handle irregular time series with a few simple but important adjustments. Based on these findings, we develop **BridgeTST**, which combines these design choices and achieves strong forecasting performance on both irregular and regular datasets. However, representative irregular-specific models often underperform on regular data. This discrepancy motivates further experiments to examine the extent to which regular and irregular forecasting can be addressed within a common framework.

Specifically, our contributions are summarized as follows:

- We establish a unified perspective on representative irregular models by organizing them around a common four-stage pipeline and analyzing their shared structure. This perspective gives the research community a clearer understanding of irregular models and provides a basis for our subsequent controlled experiments.
- Contrary to the common assumption that irregular forecasting requires a specialized architecture, our controlled experiments show that general-purpose modeling components are sufficient for effective irregular forecasting. The key is to expose the irregular observation pattern to the model. Based on this finding, we develop BridgeTST.
- Across five irregular and five regular datasets, BridgeTST achieves leading performance in both settings. We further use controlled experiments to clarify the boundary between regular and irregular forecasting, showing that irregular models underperform even in the simpler regular setting.

# 2 PRELIMINARIES

## 2.1 IRREGULAR TIME SERIES AND CANONICAL PRE-ALIGNMENT

An irregular multivariate time series contains observations collected asynchronously across multiple variables. For each variable $c \in \{1, \ldots, C\}$, let $\mathcal{O}_c = \{(t_i^{(c)}, x_i^{(c)})\}_{i=1}^{L_c}$ denote its observations. Here, $t_i^{(c)}$ and $x_i^{(c)}$ are the timestamp and value of the $i$th observation, respectively. We use *canonical pre-alignment* (CPA) to represent these observations on a shared timeline. Following the alignment strategy adopted by prior IMTS models (Zhang et al., 2024; Qiu et al., 2026), CPA constructs the timeline by sorting the union of all observation timestamps.

$$\mathcal{T} = (\tau_1, \ldots, \tau_T) = \text{sort}\left(\bigcup_{c=1}^{C} \{t_i^{(c)}\}_{i=1}^{L_c}\right). \tag{1}$$

Here, $T$ denotes the number of unique timestamps in the shared timeline. CPA then maps each observation to this timeline. This mapping produces a value matrix $\mathcal{X} \in \mathbb{R}^{T \times C}$ and a binary observation mask $\mathcal{M} \in \{0, 1\}^{T \times C}$.

$$(\mathcal{X}_{l,c}, \mathcal{M}_{l,c}) = \begin{cases} (x_i^{(c)}, 1), & \text{if } \tau_l = t_i^{(c)}, \\ (0, 0), & \text{otherwise.} \end{cases} \tag{2}$$

An entry is observed when $\mathcal{M}_{l,c} = 1$ and unobserved when $\mathcal{M}_{l,c} = 0$. The tuple $(\mathcal{X}, \mathcal{T}, \mathcal{M})$ records the observed values, their timestamps, and which variable–time pairs are observed. We characterize sampling regularity using the sampling intervals $\Delta\tau_l = \tau_{l+1} - \tau_l$. Regular time series have uniformly spaced timestamps, whereas irregular time series have non-uniformly spaced timestamps. Missing entries may occur in either setting, but irregular time series typically have sparser observation masks, and their missing ratios can differ substantially across variables.

## 2.2 PROBLEM DEFINITION

Given historical observations aligned by CPA, let $\mathbf{Q} = \langle q_f \rangle_{f=1}^{F}$ denote the $F$ future target timestamps. Let $\mathbf{Y} = [\boldsymbol{y}_f]_{f=1}^{F} \in \mathbb{R}^{F \times C}$ denote the corresponding future values, where $\boldsymbol{y}_f \in \mathbb{R}^C$ contains the values of all variables at $q_f$. The goal is to learn a forecasting model $\mathcal{F}_\theta$ that maps the historical observations and target timestamps to a forecast.

$$\hat{\mathbf{Y}} = \mathcal{F}_\theta(\mathcal{X}, \mathcal{T}, \mathcal{M}, \mathbf{Q}), \tag{3}$$

where $\hat{\mathbf{Y}} = [\hat{\boldsymbol{y}}_f]_{f=1}^{F} \in \mathbb{R}^{F \times C}$ denotes the predicted future values.

# 3 A UNIFIED PERSPECTIVE ON REGULAR AND IRREGULAR FORECASTING

Existing irregular models use substantially different architectures, making it difficult to analyze them from a unified perspective. We therefore organize their computations into a four-stage pipeline: Normalization, which adjusts the scale of the input series; Input Embedding, which maps the input series into a representation suitable for subsequent modeling; Dependency Modeling, which captures temporal and cross-variable dependencies; and Forecast Generation, which produces future values from the learned representations.

**Normalization.** We first examine normalization because instance-normalization methods rely on statistics estimated from each input series, and irregular sampling may leave some variables with too few observed values to estimate these statistics reliably. RevIN (Kim et al., 2022) is a widely used instance-normalization method for forecasting regularly sampled time series. However, most irregular models considered here do not use RevIN. TFMixer is one exception: it applies Masked RevIN on selected datasets. This variant computes normalization statistics only from observed entries and restores the original scale after forecasting. Directly transferring RevIN to irregularly sampled data may therefore be unreliable.

**Input Embedding.** Input Embedding has two parts: an embedding operator that transforms observations into model representations, and the input information provided to it. Irregular models use different embedding operators. Patch-based models such as tPatchGNN (Zhang et al., 2024) and TFMixer (Qiu et al., 2026) use TTCN to aggregate timestamped observations within temporal patches, whereas APN (Liu et al., 2026a) uses TAPA with soft temporal windows. By contrast, graph- and hypergraph-based models such as GraFITi (Yalavarthi et al., 2024) and HyperIMTS (Li et al., 2025) retain individual observations in graph or hypergraph representations. Despite these differences, these models typically use similar input information: observed values, timestamps, observation masks, and learnable variable embeddings. This information often helps the model recognize irregular sampling patterns—such as non-uniform observation times—that are difficult to infer from the observed values alone. Thus, the key to Input Embedding may lie in two aspects: the embedding operator and the input information it receives.

**Dependency Modeling.** Dependency modeling is difficult to compare because CPA-based and non-CPA-based models use different structures. CPA-based models expose the temporal and variable axes, whereas graphs and hypergraphs represent these axes implicitly. Their operators also differ—including attention, graph neural networks (GNNs), MLPs, and frequency-domain transformations—making it difficult to identify how information is propagated along these axes. To make these models comparable, we express non-CPA computations in CPA coordinates. *Surprisingly, this transformation reveals that seemingly different graph and hypergraph operations can be understood as sparse attention along the same temporal and variable axes of the CPA representation.*

To express non-CPA dependency modeling in terms of the two CPA axes, consider GraFITi. Its bipartite graph contains time nodes $\mathbf{u}_n$, variable nodes $\mathbf{v}_c$, and an edge for each observed value or forecast query. We assign each edge $(\mathbf{u}_n, \mathbf{v}_c)$ to the CPA position $(n, c)$. Let $V_{n,c}$, $T_n$, and $M_{n,c}$ denote the value, timestamp, and observation status of the edge, and let $\widetilde{\mathcal{M}} \in \{0,1\}^{N_t \times C}$ indicate which positions have corresponding edges, where $N_t$ is the number of time nodes. The initial edge representation is

$$\mathbf{h}_{n,c}^{(0)} = f_{\text{point}}(V_{n,c}, T_n, M_{n,c}) = \mathbf{z}_{n,c}. \tag{4}$$

Although GraFITi does not construct a dense grid, this mapping makes its aggregation axes explicit. Let $\mathrm{MAB}(\mathbf{q}; \mathcal{S})$ denote multi-head attention that uses $\mathbf{q}$ as the query and $\mathcal{S}$ as both keys and values. The node updates are

$$\begin{aligned} \mathbf{v}'_c &= \mathrm{MAB}\Big(\mathbf{v}_c; \big\{[\mathbf{u}_n \| \mathbf{h}_{n,c}]\big\}_{n:\widetilde{\mathcal{M}}_{n,c}=1}\Big), \\ \mathbf{u}'_n &= \mathrm{MAB}\Big(\mathbf{u}_n; \big\{[\mathbf{v}_c \| \mathbf{h}_{n,c}]\big\}_{c:\widetilde{\mathcal{M}}_{n,c}=1}\Big). \end{aligned} \tag{5}$$

Thus, $\mathbf{v}'_c$ aggregates along the time axis, whereas $\mathbf{u}'_n$ aggregates along the variable axis; alternating these updates yields sparse attention along the two CPA axes. GraFITi then updates each edge through a residual feed-forward mapping using its current state and endpoint representations. HyperIMTS admits the same interpretation through its temporal and variable hyperedges. The mapping therefore exposes a common propagation structure across these dependency modules. Whether more complex modules improve forecasting remains an empirical question.

**Forecast Generation.** Forecast Generation maps the learned representations to predictions at the requested target positions. GraFITi and HyperIMTS produce forecasts directly at the target positions, whereas tPatchGNN, APN, and TFMixer use a future-time-conditioned (FTC) prediction head that combines summaries of historical patches with future timestamp encodings. At this stage, the main distinction lies in how the encoded history is connected to the target positions.

Table 1 summarizes how these models instantiate the common pipeline.

## 4 Progressive Construction of BridgeTST

Building on the four-stage pipeline in Section 3, we progressively evaluate the components of a standard patch-based model comprising RevIN, Input embedding, temporal and variable self-attention with an FFN, and a forecast head. At each stage, we vary one component while holding the others

| Model | Normalization | Input embedding | Dependency modeling | Forecast generation |
|---|---|---|---|---|
| GraFITi (Yalavarthi et al., 2024) | No | Linear($\mathcal{X}$, $\mathcal{M}$) + time and variable embeddings | T: Attention; S: Attention | Linear |
| HyperIMTS (Li et al., 2025) | No | Linear($\mathcal{X}$, $\mathcal{M}$) + time and variable embeddings | T: Attention; S: Attention; ST: Attention | Linear |
| tPatchGNN (Zhang et al., 2024) | No | TTCN($\mathcal{X}$, $\mathcal{T}$, $\mathcal{M}$) | T: Attention; S: GNN | FTC Head |
| APN (Liu et al., 2026a) | No | TAPA($\mathcal{X}$, $\mathcal{T}$, $\mathcal{M}$) | T: Attention | FTC Head |
| TFMixer (Qiu et al., 2026) | Optional | TTCN($\mathcal{X}$, $\mathcal{T}$, $\mathcal{M}$) + variable embedding | T: Attention / MLP / NUDFT; S: MLP | FTC Head + iNUDFT |

Table 1: Representative irregular models organized by the common pipeline. T, S, and ST denote temporal, cross-variable, and joint dependencies.

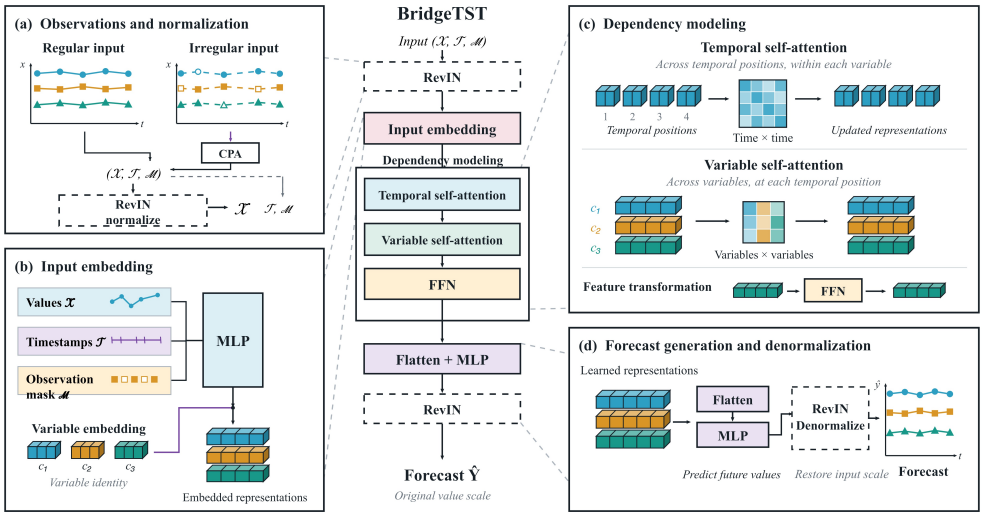


Figure 2: BridgeTST integrates progressively evaluated design choices for normalization, input embedding, dependency modeling, and forecasting. Panels (a)–(d) show CPA alignment, MLP embedding, temporal and cross-variable modeling, and an MLP forecast head. RevIN modules indicate optional input normalization and output denormalization.

fixed and carry the selected configuration forward. This process identifies design choices for effective irregular forecasting and yields **BridgeTST**, whose architecture is summarized in Figure 2.

**Experimental Setup.** We evaluate the four pipeline components on five irregular datasets spanning activity, climate, and healthcare applications. All five datasets contain sparse and asynchronous observations: HumanActivity (Vidulin et al., 2010; Yoon et al., 2019), USHCN (Menne et al., 2016), PhysioNet 2012 (P12) (Silva et al., 2012), MIMIC-III (Johnson et al., 2016), and MIMIC-IV (Johnson et al., 2023).

### 4.1 Normalization

Building on the concern raised in Section 3, we first examine the normalization component for irregularly sampled data. The key question is whether instance normalization remains reliable when observations are sparse and asynchronous.

For the normalization component, we consider three configurations: standard RevIN, Masked RevIN, and no normalization. (i) **Standard RevIN** computes instance statistics over all positions on the CPA grid, including zero-filled unobserved positions, whereas (ii) **Masked RevIN** computes these statistics only from observed entries (Kim et al., 2022).

As shown in Table 2, RevIN provides no consistent benefit across these datasets. Masked RevIN may seem like a natural remedy because it avoids the distortion that zero-filled values introduce into the normalization statistics. Yet the model performs better without normalization in most cases. These results expose an overlooked issue in prior research: regular models are often evaluated on irregular data with RevIN still enabled. RevIN is a plug-in component, not part of the forecasting

backbone. When the observation pattern is not well suited to RevIN, keeping it in the model can hurt performance, and the resulting degradation is often attributed to the model itself. A regular model may therefore appear unable to handle irregular observations when the actual problem lies in the normalization step. Thus, normalization based on instance statistics may itself be problematic when observations are sparse. Because we do not yet have a reliable way to estimate such statistics in this setting, we use no normalization in the following stages.

| Variant | HumanActivity | | USHCN | | P12 | | MIMIC-III | | MIMIC-IV | |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| RevIN | 0.0480 | 0.1299 | 0.1462 | 0.2251 | 0.3330 | 0.3761 | 0.4732 | 0.4158 | 0.2931 | 0.3303 |
| Masked RevIN | **0.0429** | **0.1165** | 0.1479 | 0.2202 | 0.3297 | 0.3658 | 0.5361 | 0.4335 | 0.3294 | 0.3357 |
| No normalization | 0.0453 | 0.1229 | **0.1419** | **0.2186** | **0.3217** | **0.3644** | **0.4559** | **0.3992** | **0.2901** | **0.3218** |

Table 2: Normalization ablation with value-only input on five irregular datasets. Shading marks the selected configuration.

### 4.2 Input Embedding

Section 3 separates Input Embedding into two parts: the operator that maps observations to model representations and the information supplied to that operator. Existing irregular models use different embedding operators, but they typically rely on the same four types of input information: observed values, timestamps, observation masks, and learnable variable embeddings. We therefore ask two questions: which inputs allow the model to recognize irregular observations, and whether a simple MLP can replace a specialized operator once those inputs are provided.

Using the no-normalization configuration from the previous stage, we use an MLP-based patch embedding. Starting from values, we successively add the observation mask, a learnable variable embedding, and timestamps. We also evaluate TTCN with all four inputs.

Let $\mathbf{P}^V$ and $\mathbf{P}^M$ denote the value and mask patches, and let $\mathbf{P}^T$ denote the timestamp patches shared across variables. We concatenate the three inputs, project them with an MLP, and add a learnable embedding for each variable:

$$\begin{aligned}(\mathbf{P}^V, \mathbf{P}^T, \mathbf{P}^M) &= \mathrm{Patchify}(\mathcal{X}, \mathcal{T}, \mathcal{M}),\\ \mathbf{h}_{i,c}^{(0)} &= \mathrm{MLP}\big(\mathbf{P}_{i,c,:}^V \| \mathbf{P}_{i,:}^T \| \mathbf{P}_{i,c,:}^M\big) + \mathbf{e}_c^{\mathrm{var}}.\end{aligned} \tag{6}$$

The resulting embeddings form $\mathbf{H}^{(0)}$.

Table 3 shows that the complete input set gives the MLP its strongest overall performance, outperforming values alone on all five datasets. With the same input set, TTCN performs better on only a few dataset–metric pairs and provides no consistent advantage over the MLP. Together, these results point to a simple but important conclusion: for input embedding, the information provided to the model matters more than the specific operator used to process it. Specialized operators such as TTCN can still improve performance on particular datasets, but their gains do not transfer consistently across datasets. Thus, a simple MLP with fewer inductive assumptions lets the model learn how much to rely on each input source, rather than imposing a fixed interaction pattern that may be specific to a dataset.

| Embedding operator (Input information) | HumanActivity | | USHCN | | P12 | | MIMIC-III | | MIMIC-IV | |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| MLP($\mathcal{X}$) | 0.0453 | 0.1229 | 0.1419 | 0.2186 | 0.3217 | 0.3644 | 0.4559 | 0.3992 | 0.2901 | 0.3218 |
| MLP($\mathcal{X} + \mathcal{M}$) | 0.0433 | 0.1205 | 0.1420 | 0.2235 | 0.3156 | 0.3589 | 0.4529 | 0.3975 | 0.2751 | 0.3149 |
| MLP($\mathcal{X} + \mathcal{M} + \mathcal{E}$) | 0.0424 | 0.1209 | 0.1433 | 0.2187 | 0.3065 | 0.3521 | 0.4107 | 0.3641 | 0.2393 | 0.2885 |
| **MLP($\mathcal{X} + \mathcal{M} + \mathcal{E} + \mathcal{T}$)** | **0.0413** | **0.1177** | 0.1408 | 0.2117 | **0.3056** | **0.3519** | **0.3910** | **0.3608** | **0.2326** | 0.2827 |
| TTCN($\mathcal{X} + \mathcal{M} + \mathcal{E} + \mathcal{T}$) | 0.0431 | 0.1182 | **0.1370** | **0.1977** | 0.3074 | 0.3551 | 0.3971 | 0.3630 | 0.2338 | **0.2771** |

Table 3: Input-embedding ablation on five irregular datasets. $\mathcal{E}$ denotes the learnable variable embeddings, and TTCN uses all four inputs. Shading marks the selected configuration.

### 4.3 Dependency Modeling

Section 3 shows that seemingly different dependency modules can be understood as different ways of propagating information along the same temporal and variable axes. This unified view leads to a focused question: once cross-variable interaction is available, do more complex modules, such as graph neural networks (GNNs), provide any additional benefit?

To isolate the effect of cross-variable modeling, we keep temporal self-attention (TSA) and the feed-forward network fixed. We evaluate three settings: no cross-variable interaction, a GNN, and variable self-attention (VSA). With VSA, the encoder layer is

$$\mathbf{H}^{(\ell+1)} = \mathrm{FFN}\Big(\mathrm{VSA}\Big(\mathrm{TSA}\Big(\mathbf{H}^{(\ell)}\Big)\Big)\Big). \tag{7}$$

As shown in Table 4, these results suggest that the variety of dependency modules may reflect different implementations of the same underlying function, rather than a need for increasingly complex designs. In this controlled study, the key factor is whether cross-variable interaction is available: both the GNN variant and variable self-attention generally outperform the no-interaction variant, while the GNN variant does not consistently improve over variable self-attention. Thus, despite the many specialized dependency designs proposed by the community, a simple variable self-attention mechanism is sufficient to capture the cross-variable information needed in our setting.

| Variant | HumanActivity | | USHCN | | P12 | | MIMIC-III | | MIMIC-IV | |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| No interaction | 0.0422 | 0.1203 | 0.1450 | 0.2133 | 0.3107 | 0.3535 | 0.4249 | 0.3734 | 0.2439 | 0.2881 |
| GNN | 0.0416 | **0.1174** | 0.1457 | 0.2124 | 0.3077 | 0.3524 | 0.4047 | 0.3637 | 0.2355 | 0.2865 |
| Variable self-attention | **0.0413** | 0.1177 | **0.1408** | **0.2117** | **0.3056** | **0.3519** | **0.3910** | **0.3608** | **0.2326** | **0.2827** |

Table 4: Cross-variable dependency ablation on five irregular datasets with temporal self-attention and a fixed FFN. Shading marks the selected configuration.

### 4.4 Forecast Generation

For forecast generation, we evaluate three prediction heads—the FTC Head, Linear Head, and MLP Head—under the same input-embedding and dependency- modeling configurations. For the MLP Head, the forecast is computed as

$$\widehat{\mathbf{Y}} = \mathrm{MLP}\Big(\mathrm{Flatten}(\mathbf{H}^{(L)})\Big) \in \mathbb{R}^{F\times C}. \tag{8}$$

Table 5 shows that the MLP Head performs best on most dataset–metric pairs, whereas the FTC Head offers no clear advantage in this setting. The FTC Head also compresses the entire historical patch sequence into a single context token, which may discard information useful for prediction. For the remaining experiments, we use the MLP Head because it preserves all encoded patches and is more expressive than a linear projection.

| Variant | HumanActivity | | USHCN | | P12 | | MIMIC-III | | MIMIC-IV | |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| FTC Head | 0.0415 | 0.1178 | 0.1533 | 0.2124 | 0.3069 | 0.3525 | 0.3973 | 0.3605 | 0.2390 | **0.2824** |
| Linear Head | 0.0427 | 0.1213 | 0.1502 | 0.2131 | 0.3070 | 0.3521 | 0.3912 | **0.3572** | 0.2358 | 0.2834 |
| MLP Head | **0.0413** | **0.1177** | **0.1408** | **0.2117** | **0.3056** | **0.3519** | **0.3910** | 0.3608 | **0.2326** | 0.2827 |

Table 5: Forecast-generation ablation evaluating three prediction heads on five irregular datasets. Shading marks the selected configuration.

## 5 Comprehensive Evaluation of BridgeTST

### 5.1 Evaluation Setup

**Datasets.** Building on the key factors identified through our controlled analysis in Section 4, we evaluate BridgeTST on the five irregular datasets and five regular long-term forecasting datasets:

| Model | Irregular | | | | | | | | | | Regular | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HumanActivity | | USHCN | | P12 | | MIMIC-III | | MIMIC-IV | | ETTh1 | | ETTm1 | | Electricity | | Traffic | | Weather | |
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| TFMixer | 0.0422 | **0.1124** | 0.1415 | **0.2019** | 0.3090 | 0.3557 | 0.4193 | 0.3794 | 0.2423 | 0.2891 | 0.7511 | 0.5596 | 0.4414 | 0.4298 | 0.8526 | 0.7028 | 1.3103 | 0.7091 | 0.2373 | 0.2727 |
| APN | 0.0421 | 0.1156 | 0.1530 | 0.2605 | 0.3132 | 0.3664 | 0.4359 | 0.4074 | 0.2570 | 0.3186 | 0.7548 | 0.6274 | 0.6972 | 0.5606 | 0.8581 | 0.7670 | 1.4145 | 0.7948 | 0.2962 | 0.3528 |
| HyperIMTS | 0.0419 | 0.1136 | 0.1839 | 0.2803 | **0.3016** | 0.3613 | 0.3982 | 0.3844 | **0.2239** | 0.2866 | N/A | | N/A | | N/A | | N/A | | N/A | |
| tPatchGNN | 0.0435 | 0.1234 | 0.1632 | 0.2600 | 0.3112 | 0.3659 | 0.4320 | 0.4063 | 0.2459 | 0.3032 | 0.7249 | 0.5995 | 0.7077 | 0.5776 | 0.8378 | 0.7576 | 1.3952 | 0.7752 | 0.2538 | 0.3148 |
| GraFITi | 0.0423 | 0.1149 | 0.1668 | 0.2837 | 0.3038 | 0.3611 | 0.4267 | 0.4030 | 0.2495 | 0.3068 | N/A | | N/A | | N/A | | N/A | | N/A | |
| TimeMixer | 0.0741 | 0.1809 | 0.1685 | 0.2535 | 0.3412 | 0.3829 | 0.5291 | 0.4318 | 0.2978 | 0.3367 | 0.4279 | 0.4377 | 0.3527 | 0.3748 | 0.1599 | 0.2496 | 0.4032 | 0.2577 | 0.2254 | 0.2551 |
| iTransformer | 0.1184 | 0.2218 | 0.2219 | 0.2801 | 0.3602 | 0.4016 | 0.5277 | 0.4499 | 0.4233 | 0.4317 | 0.4499 | 0.4609 | 0.3577 | 0.3849 | 0.1616 | 0.2541 | **0.3906** | 0.2593 | 0.2294 | 0.2613 |
| PatchTST | 0.0725 | 0.1786 | 0.1681 | 0.2558 | 0.3276 | 0.3699 | 0.5327 | 0.4221 | 0.3323 | 0.3605 | 0.4038 | 0.4290 | **0.3451** | 0.3701 | 0.1667 | 0.2614 | 0.4039 | 0.2635 | 0.2232 | 0.2512 |
| Crossformer | 0.0579 | 0.1469 | 0.1514 | 0.2183 | 0.3284 | 0.3845 | 0.4760 | 0.4379 | 0.3112 | 0.3683 | 0.4305 | 0.4530 | 0.4456 | 0.4264 | 0.1742 | 0.2700 | 0.5114 | 0.2801 | 0.2284 | 0.2758 |
| DLinear | 0.2287 | 0.3081 | 0.3463 | 0.3791 | 0.4242 | 0.4547 | 0.6972 | 0.5626 | 0.4996 | 0.4893 | 0.4216 | 0.4306 | 0.3504 | 0.3694 | 0.1631 | 0.2547 | 0.4287 | 0.2734 | 0.2392 | 0.2736 |
| BridgeTST | **0.0413** | 0.1177 | **0.1408** | 0.2117 | 0.3056 | **0.3519** | **0.3910** | **0.3608** | 0.2326 | **0.2827** | **0.4010** | **0.4188** | 0.3489 | **0.3671** | **0.1561** | **0.2483** | 0.4032 | **0.2471** | **0.2138** | **0.2429** |

Table 6: Forecasting results on five irregular and five regular datasets. Regular results are averaged over four horizons; shading, bold, and underlining mark BridgeTST, best, and second-best results. N/A denotes infeasible evaluations.

ETTh1, ETTm1, Electricity, Traffic, and Weather. All irregular datasets use an 8:1:1 chronological split. HumanActivity uses a 3,000-millisecond lookback and a 300-millisecond forecast window. USHCN uses three years of history and predicts the next three target timestamps, while each clinical dataset uses 36 hours of history and predicts the next three target timestamps. For the regular datasets, we use a 512-step lookback and forecast horizons of $\{96, 192, 336, 720\}$. ETTh1 and ETTm1 use 6:2:2 chronological splits, whereas Electricity, Traffic, and Weather use 7:1:2 splits.

**Baselines.** We evaluate BridgeTST alongside five irregular models—GraFITi, tPatchGNN, HyperIMTS, APN, and TFMixer—and five regular models—DLinear, PatchTST, Crossformer, iTransformer, and TimeMixer—using MSE and MAE. Regular results are averaged over the four forecast horizons. For BridgeTST, RevIN is disabled for irregular datasets and enabled for regular datasets.

**Implementation details.** BridgeTST and its variants use the L1 loss, whereas the baselines follow their published training objectives. All models are trained with Adam (Kingma & Ba, 2015) using three random seeds. We do not use the "Drop Last" option. Results are averaged over seeds using the same data splits and dataset-specific forecast horizons. Further implementation details are provided in Appendix A.3.

### 5.2 Main Results

Table 6 reports MSE and MAE for BridgeTST and Baselines on five irregular and five regular datasets under the evaluation protocols described above. BridgeTST achieves strong forecasting performance in both settings. On the irregular datasets, it achieves the best result on at least one of the two metrics for every dataset. On the regular datasets, it achieves the best result on eight of the ten dataset–metric pairs and remains competitive on the rest.

**A common architecture for both settings.** These results answer the question posed in the Introduction Section: *a model developed for regularly sampled data can also handle irregular observations once we identify the choices that matter for irregular observations*. BridgeTST combines these choices in a single model for both irregular and regular data. Regular forecasting can be viewed as the simpler case of the same problem, with evenly spaced timestamps and dense observations; BridgeTST therefore also performs well in the regular setting.

**Limitations of irregular-specific models.** By contrast, the irregular models evaluated above do not perform as well as BridgeTST on regular datasets: some perform substantially worse, while others cannot be evaluated because constructing observation-level graphs is computationally expensive. If regular data can be viewed as a simpler case of irregular data, why do models that handle irregular observations nevertheless perform poorly on regular data?

### 5.3 Why Do Irregular Models Underperform on Regular Data?

BridgeTST provides a natural starting point for investigating this question. It incorporates the information needed to handle irregular observations and performs well on regular data. BridgeTST differs from the irregular models evaluated above in one important respect: it does not include mod-

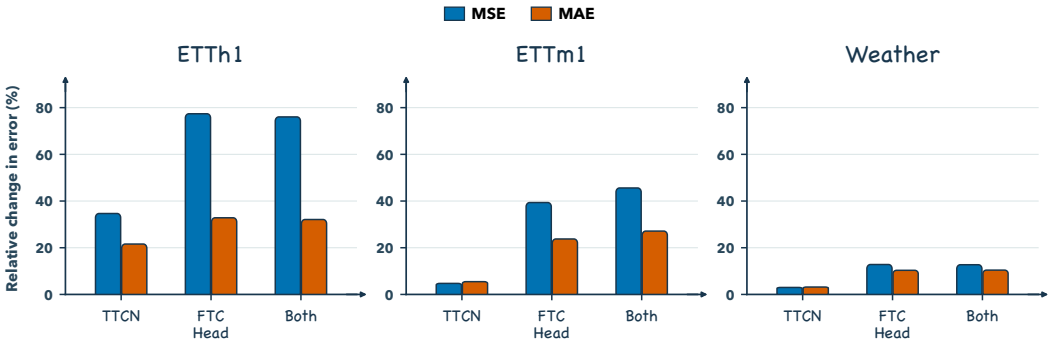


Figure 3: Relative MSE and MAE changes on three representative regular datasets: TTCN replaces the input-embedding MLP, the FTC Head replaces the forecast-generation MLP, or both. Positive values indicate degradation.

ules designed specifically for irregular forecasting. We therefore hypothesize that these modules may contribute to the poor performance of irregular models on regular data. To examine this possibility, we focus on two commonly used modules from different stages of the pipeline: TTCN for input embedding and the FTC Head for forecast generation.

We use BridgeTST with its original MLP-based input embedding and MLP forecast head as the baseline. On the three regular datasets shown in Figure 3, we evaluate three variants: replacing the input-embedding MLP with TTCN, replacing the forecast-generation MLP with the FTC Head, and replacing both components simultaneously. All other model components are kept unchanged.

The results support this hypothesis. Across the three representative regular datasets shown in Figure 3, each single-component replacement causes clear performance degradation relative to the original BridgeTST configuration. The FTC Head consistently produces a larger increase in error than TTCN. Replacing both components also performs worse and provides no consistent improvement over the FTC-only variant.

These results point to a more specific explanation. The issue is not whether the model uses sampling information. It is whether the forecasting process becomes too tightly tied to a particular sampling pattern. TTCN uses each observation's value and timestamp to generate time-aware filter weights and uses them to aggregate the valid observations within a patch. The FTC Head first compresses the historical patch sequence into a context vector and then conditions the forecast on future timestamps. These mechanisms can help when irregular sampling is central to the task, but they also tie the forecasting process directly to the sampling pattern. BridgeTST instead provides sampling information to a simple MLP and lets the data determine how much it should matter. This flexibility helps explain its strong performance in both irregular and regular settings.

## 6 Conclusion

Representative irregular models use substantially different architectures. We organize and analyze these models using a common four-stage pipeline, helping the community better understand irregular models. Building on this organization, controlled experiments show that effective irregular forecasting does not require a specialized architecture. General-purpose modeling components are sufficient when the forecasting pipeline is appropriately adapted to irregular observations. Based on this finding, we construct BridgeTST.

Rather than pursuing another specialized state-of-the-art model, we develop BridgeTST to examine whether a single model can support both regular and irregular settings. It achieves leading performance across five irregular and five regular datasets. Yet irregular models underperform even in the simpler regular setting. We then use additional controlled experiments to investigate why irregular models underperform on regular data.

### AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing and code development and debugging. All AI-assisted text and code were carefully reviewed and verified by the authors. The research questions, methodology, experimental design, analysis, and scientific claims were developed independently by the authors. We take full responsibility for the final content of this work.

### ETHICS STATEMENT

This work focuses on methodological research for irregular and regular time series forecasting. Our experiments use existing benchmark datasets; we do not collect new human-subject data or intentionally collect personally identifiable information. The clinical benchmark datasets are used in their existing form. The proposed method is intended for general-purpose forecasting research, and we are not aware of any direct ethical risks arising from the methodology itself.

### REPRODUCIBILITY STATEMENT

The four-stage perspective and BridgeTST are described in Sections 3–4. Section 5 presents the experimental setup and main evaluation, and Section 7 provides dataset statistics, baseline descriptions, implementation details, target-mask and hyperparameter sensitivity, and per-horizon results. The benchmark datasets are available from their original sources; the source code and training and evaluation scripts will be released.

## REFERENCES

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. Wiley, 2015.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David A. Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8:6085, 2018. doi: 10.1038/s41598-018-24271-9.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems*, volume 32, pp. 7379–7390, 2019.

James Douglas Hamilton. *Time series analysis*. Princeton University Press, 2020.

Max Horn, Michael Moor, Christian Bock, Bastian Rieck, and Karsten M. Borgwardt. Set functions for time series. In *ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4353–4363, 2020.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1, 2023. doi: 10.1038/s41597-022-01899-x.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3(1):160035, 2016. doi: 10.1038/sdata.2016.35.

Patrick Kidger, James Morrill, James Foster, and Terry J. Lyons. Neural controlled differential equations for irregular time series. In *NeurIPS 2020*, 2020.

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

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 95–104, 2018. doi: 10.1145/3209978.3210006.

Boyuan Li, Yicheng Luo, Zhen Liu, Junhao Zheng, Jianming Lv, and Qianli Ma. HyperIMTS: Hypergraph neural network for irregular multivariate time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35502–35518. PMLR, 2025.

Jiexi Liu, Meng Cao, and Songcan Chen. TimeCHEAT: A channel harmony strategy for irregularly sampled multivariate time series analysis. In *AAAI 2025*, pp. 18861–18869, 2025a.

Xu Liu, Juncheng Liu, Gerald Woo, Taha Aksu, Yuxuan Liang, Roger Zimmermann, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Moirai-MoE: Empowering time series foundation models with sparse mixture of experts. In *International Conference on Machine Learning*, 2025b.

Xvyuan Liu, Xiangfei Qiu, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Jilin Hu, and Bin Yang. Rethinking irregular time series forecasting: A simple yet effective baseline. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 23873–23881, 2026a. doi: 10.1609/aaai.v40i28.39563.

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

Yong Liu, Xingjian Su, Shiyu Wang, Haoran Zhang, Haixuan Liu, Yuxuan Wang, Zhou Ye, Yang Xiang, Jianmin Wang, and Mingsheng Long. Timer-S1: A billion-scale time series foundation model with serial scaling. *arXiv preprint arXiv:2603.04791*, 2026b.

Matthew J. Menne, Claude N. Williams, Jr., and Russell S. Vose. Long-term daily and monthly climate records from stations across the contiguous United States (U.S. Historical Climatology Network). Technical report, Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, 2016.

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

Xiangfei Qiu, Kangjia Yan, Xvyuan Liu, Xingjian Wu, and Jilin Hu. Bridging time and frequency: A joint modeling framework for irregular multivariate time series forecasting. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/65148.

Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. Predicting in-hospital mortality of ICU patients: The PhysioNet/computing in cardiology challenge 2012. *Computing in Cardiology*, 39:245–248, 2012.

Vedrana Vidulin, Mitja Luštrek, Boštjan Kaluža, Rok Piltaver, and Jana Krivec. Localization data for person activity, 2010.

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

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *NeurIPS*, volume 34, pp. 22419–22430, 2021.

Vijaya Krishna Yalavarthi, Kiran Madhusudhanan, Randolf Scholz, Nourhan Ahmed, Johannes Burchert, Shayan Jawed, Stefan Born, and Lars Schmidt-Thieme. GraFITi: Graphs for forecasting irregularly sampled time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16255–16263, 2024. doi: 10.1609/aaai.v38i15.29560.

Jinsung Yoon, William R Zame, and Mihaela van der Schaar. Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Transactions on Biomedical Engineering*, 66(5):1477–1490, 2019. doi: 10.1109/TBME.2018.2874712.

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. doi: 10.1609/aaai.v37i9.26317.

Jiawen Zhang, Shun Zheng, Wei Cao, Jiang Bian, and Jia Li. Warpformer: A multi-scale modeling approach for irregular clinical time series. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3273–3285, 2023. doi: 10.1145/3580305.3599543.

Weijia Zhang, Chenlong Yin, Hao Liu, Xiaofang Zhou, and Hui Xiong. Irregular multivariate time series forecasting: A transformable patching graph neural networks approach. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 60179–60196. PMLR, 2024.

Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. In *ICLR 2022*, 2022.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=vSVLM2j9eie.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11106–11115, 2021. doi: 10.1609/aaai.v35i12.17325.

Ziyu Zhou, Yiming Huang, Yanyun Wang, Yuankai Wu, James Kwok, and Yuxuan Liang. Revitalizing canonical pre-alignment for irregular multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 29115–29123, 2026. doi: 10.1609/aaai.v40i34.40149.

# A Experimental Details

## A.1 Datasets

To evaluate forecasting under irregular and regular sampling, we use ten multivariate time-series datasets spanning activity recognition, climate, healthcare, energy, and transportation. The five irregular datasets contain sparse, asynchronous observations, whereas the five regular datasets are standard long-term forecasting benchmarks defined on regular temporal grids. Table 7 summarizes their statistics and forecasting settings.

### A.1.1 Irregular Datasets

We follow the preprocessing and forecasting protocol of HyperIMTS (Li et al., 2025) for the irregular datasets. Each dataset is split into training, validation, and test sets in an 8:1:1 ratio. The lookback windows are 3,000 milliseconds for HumanActivity, three years for USHCN, and 36 hours for each clinical dataset. HumanActivity uses a 300-millisecond forecast window; the other irregular datasets use a three-step forecast horizon.

- **HumanActivity** (Vidulin et al., 2010; Yoon et al., 2019) contains three-dimensional positional measurements collected by wearable sensors during human activities. In the standard processed benchmark, each instance is represented by 12 variables on a 1-millisecond grid. The processed dataset contains 1,359 instances, and 75.0% of the aligned entries are unobserved.
- **USHCN** (Menne et al., 2016; De Brouwer et al., 2019) is derived from the United States Historical Climatology Network and contains daily climate measurements collected at weather stations across the United States. We use the standard subset of 1,114 stations and five climate variables adopted in prior irregular forecasting studies. On the daily grid, 77.9% of the aligned entries are unobserved.
- **P12** (Silva et al., 2012) comprises multivariate clinical measurements collected during intensive-care-unit stays. Following prior preprocessing, we use 11,981 patient instances and 36 variables, with measurements aligned to a one-hour grid. In the resulting benchmark, 85.7% of the aligned entries are unobserved.
- **MIMIC-III** (Johnson et al., 2016) is an ICU database containing heterogeneous clinical measurements recorded at irregular times. The processed forecasting benchmark contains 21,250 patient instances and 96 variables aligned to a 30-minute grid. In this benchmark, 94.2% of the aligned entries are unobserved, making it substantially sparser than P12.
- **MIMIC-IV** (Johnson et al., 2023) is a recent critical-care database with clinical variables recorded at irregular times. We use the processed benchmark with 17,874 patient instances and 100 variables on a one-minute grid. It has the highest unobserved rate among our datasets: 97.8%.

### A.1.2 Regular Datasets

The regular datasets follow the standard long-term forecasting protocol. We use a fixed lookback window of 512 and forecast horizons of 96, 192, 336, and 720. ETTh1 and ETTm1 use chronological splits of 6:2:2, whereas Electricity, Traffic, and Weather use splits of 7:1:2.

- **ETTh1** (Zhou et al., 2021) records seven variables from electricity transformers, including oil temperature and load-related measurements, at hourly intervals. The benchmark contains 14,400 timestamps. We evaluate the four standard long-term forecasting horizons with the same 512-step lookback window.
- **ETTm1** (Zhou et al., 2021) contains the same seven variables as ETTh1, recorded every 15 minutes. This higher sampling frequency yields 57,600 timestamps. We use the same lookback window and forecast horizons as for ETTh1.
- **Electricity** (Lai et al., 2018) records hourly electricity consumption for 321 clients. The benchmark contains 26,304 timestamps and uses the standard chronological split. Each variable corresponds to one client's consumption series.

- **Traffic** (Wu et al., 2021) contains hourly road-occupancy measurements collected by 862 sensors. It contains 17,544 timestamps and has the largest number of variables among the ten datasets. This high dimensionality also makes observation-level graph construction expensive for graph-based irregular models.
- **Weather** (Wu et al., 2021) contains 21 meteorological variables recorded every 10 minutes. It contains 52,696 timestamps covering multiple weather indicators. We use the same long-term forecasting protocol as for Electricity and Traffic.

Table 7: Dataset statistics and forecasting settings. "Samples" and "Timestamps" denote the number of instances and the temporal length, respectively. "Lookback" and "Forecast" denote the historical input range and future prediction range, respectively.

| Setting | Dataset | Domain | Samples / Timestamps | Variables | Unobserved rate | Resolution | Split | Lookback | Forecast |
|---|---|---|---|---|---|---|---|---|---|
| Irregular | HumanActivity | Activity | 1,359 | 12 | 75.0% | 1 ms | 8:1:1 | 3,000 ms | 300 ms |
| | USHCN | Climate | 1,114 | 5 | 77.9% | 1 day | 8:1:1 | 3 years | 3 steps |
| | P12 | Healthcare | 11,981 | 36 | 85.7% | 1 hour | 8:1:1 | 36 hours | 3 steps |
| | MIMIC-III | Healthcare | 21,250 | 96 | 94.2% | 30 min | 8:1:1 | 36 hours | 3 steps |
| | MIMIC-IV | Healthcare | 17,874 | 100 | 97.8% | 1 min | 8:1:1 | 36 hours | 3 steps |
| Regular | ETTh1 | Energy | 14,400 | 7 | 0% | 1 hour | 6:2:2 | 512 | 96/192/336/720 |
| | ETTm1 | Energy | 57,600 | 7 | 0% | 15 min | 6:2:2 | 512 | 96/192/336/720 |
| | Electricity | Energy | 26,304 | 321 | 0% | 1 hour | 7:1:2 | 512 | 96/192/336/720 |
| | Traffic | Transportation | 17,544 | 862 | 0% | 1 hour | 7:1:2 | 512 | 96/192/336/720 |
| | Weather | Climate | 52,696 | 21 | 0% | 10 min | 7:1:2 | 512 | 96/192/336/720 |

### A.2 Baselines

We evaluate BridgeTST alongside regular models and irregular models.

**DLinear** (Zeng et al., 2023) decomposes each time series into trend and seasonal components and uses linear layers for forecasting.

**PatchTST** (Nie et al., 2023) divides each sequence into fixed-length patches and encodes them with a Transformer encoder using channel-independent processing.

**Crossformer** (Zhang & Yan, 2023) embeds a multivariate time series as a two-dimensional array of dimension–segment representations and uses two-stage attention to capture dependencies across time and variables.

**iTransformer** (Liu et al., 2024) reverses the standard attention pattern by applying self-attention across variables and feed-forward networks across time steps.

**TimeMixer** (Wang et al., 2024) decomposes time series at multiple scales and mixes information across scales via past-decomposable mixing.

**GraFITi** (Yalavarthi et al., 2024) represents observations and prediction queries as nodes in a bipartite graph and uses graph attention for forecasting.

**tPatchGNN** (Zhang et al., 2024) organizes irregular observations into temporal patches and models dependencies across variables with graph neural networks.

**HyperIMTS** (Li et al., 2025) represents observations and forecast targets as nodes of a hypergraph and models temporal and cross-variable dependencies through node–hyperedge message passing.

**APN** (Liu et al., 2026a) uses Time-Aware Patch Aggregation to convert irregular observations into patch representations.

**TFMixer** (Qiu et al., 2026) jointly models temporal and frequency-domain patterns in irregular data.

GraFITi and HyperIMTS are not evaluated on the regular datasets because, under our resource budget, constructing their observation-level graph or hypergraph representations is computationally infeasible for these dense benchmarks.

### A.3 Implementation Details

For BridgeTST, we disable RevIN for irregular datasets and enable it for regular datasets.

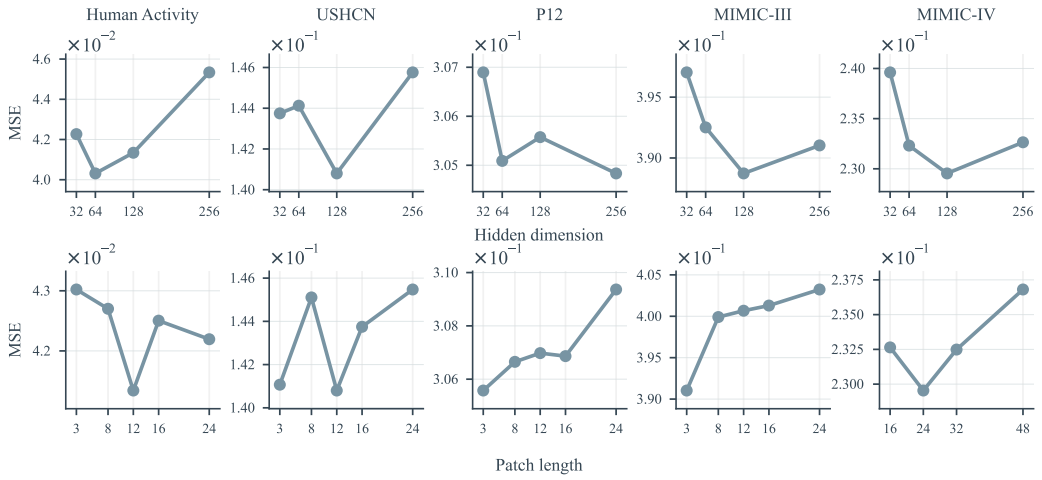


Figure 4: Sensitivity to hidden dimension and patch length on five irregular datasets. Lower MSE indicates better performance.

For GraFITi and HyperIMTS, the target-side mask used to construct prediction queries differs from the realized target mask used for loss computation and evaluation. During training and inference, we evaluate two target-side mask settings: the realized target-availability mask (R) and an all-ones mask (O). The first and second letters denote the training and inference masks, respectively. Because future-target availability is not assumed to be known when prediction queries are constructed at test time, we use the OO configuration as the main setting. In all configurations, we retain the realized target mask for loss computation and evaluation. Table 8 reports the corresponding sensitivity results.

Table 8: Target-mask configurations for GraFITi and HyperIMTS. R and O denote the realized target-availability mask and the all-ones mask, respectively. The first and second letters indicate the training and inference masks. Lower values indicate better performance.

| Dataset | GraFITi | | | | | | | | HyperIMTS | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RR | | OR | | RO | | OO | | RR | | OR | | RO | | OO | |
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| HumanActivity | 0.0419 | 0.1144 | 0.0439 | 0.1215 | 0.0560 | 0.1572 | 0.0423 | 0.1149 | 0.0416 | 0.1134 | 0.0438 | 0.1219 | 0.0547 | 0.1564 | 0.0419 | 0.1136 |
| USHCN | 0.1688 | 0.2866 | 0.1683 | 0.2837 | 0.1620 | 0.2638 | 0.1668 | 0.2837 | 0.1870 | 0.3090 | 0.1899 | 0.2902 | 0.4604 | 0.3467 | 0.1839 | 0.2803 |
| P12 | 0.3041 | 0.3603 | 0.3096 | 0.3654 | 0.3113 | 0.3747 | 0.3038 | 0.3611 | 0.3014 | 0.3621 | 0.3175 | 0.3772 | 0.3035 | 0.3636 | 0.3016 | 0.3613 |
| MIMIC-III | 0.3687 | 0.3704 | 0.4297 | 0.4011 | 0.4905 | 0.4424 | 0.4267 | 0.4030 | 0.3588 | 0.3666 | 0.4982 | 0.4743 | 0.4825 | 0.4415 | 0.3982 | 0.3844 |
| MIMIC-IV | 0.2358 | 0.3023 | 0.2557 | 0.3120 | 0.2810 | 0.3262 | 0.2495 | 0.3068 | 0.2148 | 0.2790 | 0.2621 | 0.3251 | 0.2914 | 0.3404 | 0.2239 | 0.2866 |

### A.4 Hyperparameter Sensitivity

We analyze two key model hyperparameters: the hidden dimension $d_{\text{model}}$, which controls model capacity, and the patch length, which determines the granularity of temporal aggregation. Figure 4 shows the results.

The preferred values vary across datasets. A moderate hidden dimension is generally sufficient: $d_{\text{model}} = 128$ achieves the best or near-best performance on most datasets, whereas larger dimensions can degrade performance, especially on smaller datasets where extra capacity may cause overfitting. Patch length is more dataset-dependent. Short-to-moderate patches generally perform well, whereas overly long patches tend to hurt performance, particularly on the MIMIC datasets. This pattern reflects a trade-off between temporal resolution and contextual coverage. Shorter patches preserve finer-grained dynamics, whereas longer patches cover broader context but may blur local variation. These results suggest selecting both hyperparameters according to dataset characteristics rather than fixing them across all datasets.

### A.5 PER-HORIZON RESULTS ON REGULAR DATASETS

The main text reports results on regular datasets averaged over four forecast horizons. Table 9 reports the results for each horizon and their average. GraFITi and HyperIMTS are omitted because, under our resource budget, constructing their observation-level graph or hypergraph representations is computationally infeasible on these dense regular benchmarks.

Table 9: Per-horizon and average forecasting results on the five regular datasets. Lower MSE and MAE indicate better performance. Avg. denotes the average over forecast horizons $\{96, 192, 336, 720\}$. Bold and underlined values indicate the best and second-best results among methods with reported results.

| Dataset | Horizon | BridgeTST | | TFMixer | | APN | | tPatchGNN | | TimeMixer | | iTransformer | | PatchTST | | Crossformer | | DLinear | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | 96 | 0.3650 | 0.3892 | 0.7417 | 0.5288 | 0.7097 | 0.5788 | 0.6974 | 0.5620 | 0.3748 | 0.4008 | 0.4013 | 0.4225 | **0.3552** | 0.3899 | 0.3876 | 0.4269 | 0.3665 | **0.3859** |
| | 192 | 0.3987 | **0.4140** | 0.7480 | 0.5393 | 0.7464 | 0.6184 | 0.7092 | 0.5793 | 0.4205 | 0.4308 | 0.4252 | 0.4383 | **0.3962** | 0.4203 | 0.4086 | 0.4273 | 0.4107 | 0.4155 |
| | 336 | **0.4114** | **0.4238** | 0.7595 | 0.5619 | 0.7479 | 0.6294 | 0.7218 | 0.5981 | 0.4386 | 0.4399 | 0.4398 | 0.4526 | 0.4163 | 0.4352 | 0.4410 | 0.4514 | 0.4424 | 0.4383 |
| | 720 | **0.4289** | **0.4483** | 0.7553 | 0.6081 | 0.8150 | 0.6831 | 0.7712 | 0.6584 | 0.4779 | 0.4792 | 0.5334 | 0.5303 | 0.4475 | 0.4704 | 0.4846 | 0.5062 | 0.4668 | 0.4828 |
| | Avg. | **0.4010** | **0.4188** | 0.7511 | 0.5596 | 0.7548 | 0.6274 | 0.7249 | 0.5995 | 0.4279 | 0.4377 | 0.4499 | 0.4609 | 0.4038 | 0.4290 | 0.4305 | 0.4530 | 0.4216 | 0.4306 |
| ETTm1 | 96 | **0.2818** | **0.3281** | 0.3441 | 0.3783 | 0.6771 | 0.5451 | 0.6823 | 0.5549 | 0.2906 | 0.3398 | 0.2998 | 0.3493 | 0.2824 | 0.3320 | 0.3191 | 0.3735 | 0.2942 | 0.3357 |
| | 192 | 0.3292 | **0.3557** | 0.3762 | 0.3978 | 0.6911 | 0.5553 | 0.6966 | 0.5676 | 0.3327 | 0.3657 | 0.3355 | 0.3715 | **0.3255** | 0.3586 | 0.3497 | 0.3866 | 0.3300 | 0.3563 |
| | 336 | 0.3632 | 0.3767 | 0.5109 | 0.4640 | 0.7023 | 0.5639 | 0.7094 | 0.5832 | 0.3649 | 0.3800 | 0.3680 | 0.3913 | **0.3580** | 0.3788 | 0.3926 | 0.4064 | 0.3612 | **0.3766** |
| | 720 | 0.4215 | **0.4079** | 0.5346 | 0.4790 | 0.7183 | 0.5782 | 0.7425 | 0.6049 | 0.4225 | 0.4136 | 0.4275 | 0.4274 | **0.4144** | 0.4110 | 0.7210 | 0.5391 | 0.4160 | 0.4089 |
| | Avg. | 0.3489 | **0.3671** | 0.4414 | 0.4298 | 0.6972 | 0.5606 | 0.7077 | 0.5776 | 0.3527 | 0.3748 | 0.3577 | 0.3849 | **0.3451** | 0.3701 | 0.4456 | 0.4264 | 0.3504 | 0.3694 |
| Electricity | 96 | **0.1267** | **0.2158** | 0.6520 | 0.6045 | 0.8347 | 0.7579 | 0.8083 | 0.7452 | 0.1295 | 0.2211 | 0.1320 | 0.2261 | 0.1360 | 0.2342 | 0.1363 | 0.2333 | 0.1364 | 0.2294 |
| | 192 | **0.1429** | **0.2343** | 0.8889 | 0.7205 | 0.8463 | 0.7620 | 0.8131 | 0.7435 | 0.1477 | 0.2373 | 0.1505 | 0.2434 | 0.1529 | 0.2499 | 0.1525 | 0.2521 | 0.1505 | 0.2419 |
| | 336 | **0.1599** | 0.2559 | 0.9279 | 0.7399 | 0.8634 | 0.7704 | 0.8509 | 0.7648 | 0.1634 | **0.2541** | 0.1656 | 0.2594 | 0.1693 | 0.2649 | 0.1720 | 0.2759 | 0.1655 | 0.2580 |
| | 720 | **0.1948** | 0.2872 | 0.9417 | 0.7462 | 0.8880 | 0.7776 | 0.8787 | 0.7771 | 0.1990 | **0.2859** | 0.1984 | 0.2874 | 0.2084 | 0.2966 | 0.2360 | 0.3187 | 0.1998 | 0.2893 |
| | Avg. | **0.1561** | **0.2483** | 0.8526 | 0.7028 | 0.8581 | 0.7670 | 0.8378 | 0.7576 | 0.1599 | 0.2496 | 0.1616 | 0.2541 | 0.1667 | 0.2614 | 0.1742 | 0.2700 | 0.1631 | 0.2547 |
| Traffic | 96 | 0.3644 | **0.2265** | 1.1248 | 0.6034 | 1.3901 | 0.7914 | 1.3194 | 0.7512 | 0.3688 | 0.2343 | **0.3615** | 0.2463 | 0.3767 | 0.2505 | 0.4724 | 0.2631 | 0.4106 | 0.2639 |
| | 192 | 0.4073 | **0.2403** | 1.3054 | 0.7049 | 1.4029 | 0.7947 | 1.3738 | 0.7681 | 0.3937 | 0.2631 | **0.3826** | 0.2551 | 0.3939 | 0.2579 | 0.5093 | 0.2753 | 0.4193 | 0.2674 |
| | 336 | 0.4055 | 0.2462 | 1.3599 | 0.7393 | 1.4197 | 0.7951 | 1.4229 | 0.7847 | 0.3974 | **0.2432** | **0.3960** | 0.2628 | 0.4029 | 0.2620 | 0.5048 | 0.2773 | 0.4262 | 0.2718 |
| | 720 | 0.4355 | 0.2756 | 1.4512 | 0.7889 | 1.4452 | 0.7981 | 1.4645 | 0.7967 | 0.4528 | 0.2901 | **0.4222** | **0.2729** | 0.4420 | 0.2835 | 0.5592 | 0.3046 | 0.4589 | 0.2907 |
| | Avg. | 0.4032 | **0.2471** | 1.3103 | 0.7091 | 1.4145 | 0.7948 | 1.3952 | 0.7752 | 0.4032 | 0.2577 | **0.3906** | 0.2593 | 0.4039 | 0.2635 | 0.5114 | 0.2801 | 0.4287 | 0.2734 |
| Weather | 96 | **0.1372** | **0.1753** | 0.1659 | 0.2181 | 0.2264 | 0.2991 | 0.1741 | 0.2476 | 0.1463 | 0.1884 | 0.1550 | 0.1977 | 0.1457 | 0.1852 | 0.1429 | 0.2100 | 0.1718 | 0.2124 |
| | 192 | **0.1795** | **0.2197** | 0.2017 | 0.2471 | 0.2712 | 0.3354 | 0.2251 | 0.2952 | 0.1917 | 0.2319 | 0.1994 | 0.2396 | 0.1894 | 0.2272 | 0.1863 | 0.2303 | 0.2125 | 0.2515 |
| | 336 | **0.2311** | **0.2605** | 0.2601 | 0.2915 | 0.3178 | 0.3695 | 0.2781 | 0.3414 | 0.2410 | 0.2738 | 0.2449 | 0.2784 | 0.2425 | 0.2692 | 0.2496 | 0.2846 | 0.2555 | 0.2896 |
| | 720 | **0.3074** | **0.3162** | 0.3214 | 0.3340 | 0.3696 | 0.4074 | 0.3378 | 0.3749 | 0.3226 | 0.3263 | 0.3183 | 0.3297 | 0.3153 | 0.3231 | 0.3349 | 0.3783 | 0.3170 | 0.3408 |
| | Avg. | **0.2138** | **0.2429** | 0.2373 | 0.2727 | 0.2962 | 0.3528 | 0.2538 | 0.3148 | 0.2254 | 0.2551 | 0.2294 | 0.2613 | 0.2232 | 0.2512 | 0.2284 | 0.2758 | 0.2392 | 0.2736 |