# REPACKAGING TEMPORAL EVIDENCE: A UNIFYING INTERFACE FOR TEMPORAL PREDICTION

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

## ABSTRACT

Temporal tabular prediction and irregular time-series prediction often arise from similar longitudinal records, but they expose different evidence at query time. Temporal tables usually present a timestamped covariate row and evaluate generalization to future periods, whereas irregular time-series protocols present entity-specific observations and query values at selected times. We study this gap as a protocol issue rather than a modality boundary. We introduce a unified prediction interface that specifies the query-time information set and the prediction rule applied to it, and use it to diagnose protocol mismatches in cross-domain reuse. The interface highlights two common failures: entity collapse, where a temporal table is forced into a single sequence across unrelated entities, and temporal relation mismatch, where tabulation hides the within-sequence structure expected by a time-series model. Empirically, we use the interface as a diagnostic tool in both directions. A retrieval-based tabular predictor can be evaluated on irregular time-series tasks once candidate sets are repackaged as temporally embedded, entity-specific contexts. Conversely, selected temporal tables admit time-series evaluations when recoverable entity histories or source sequences are restored before modeling. These results highlight repackaging temporal evidence as a way to enable cross-domain reuse, without implying that either model family subsumes the other.

## 1 INTRODUCTION

Real-world prediction problems often unfold along a time axis: medical records accrue over months or years (Rajkomar et al., 2018), sensors report asynchronously (Ganesan et al., 2004), and business features or user populations drift with seasons and policy changes (Nederstigt et al., 2014). In such cases, models are deployed on evolving data rather than a static closed dataset (Zhou, 2022). We refer to this setting as *temporal prediction*: at query time, the model uses the information exposed by the protocol to predict the target.

Research on temporal prediction has largely progressed along two tracks that often originate from similar data sources. Figure 1 illustrates their default protocols. One track is *tabular prediction under temporal distribution shift* (Rubachev et al., 2025), which treats each timestamped record as a covariate-rich instance and emphasizes heterogeneous feature interactions and future-period generalization (Gardner et al., 2023; Cai & Ye, 2025a;b). The other is *irregular time-series prediction*, where unevenly sampled and asynchronous observations are organized as histories for interpolation, extrapolation, or classification (Box et al., 2015; Moebus et al., 2025). This has motivated time-aware recurrent models (Che et al., 2018), continuous-time latent dynamics (Rubanova et al., 2019), and attention-based query-time interpolation (Shukla & Marlin, 2021).

The two tracks bring complementary inductive biases: time-series methods model time-indexed evidence and irregular histories, while tabular methods model heterogeneous covariates and feature interactions. Yet direct comparison is difficult because benchmarks differ not only in model class but also in what information is available at query time. This paper places the two settings into a common protocol view, with the narrower goal of diagnosing when cross-domain reuse is well specified.

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.

Our starting point is that much of the divide is induced by how temporal evidence is packaged. The same longitudinal record can be presented as a table of timestamped rows, a set of labeled candidates, an entity-specific history window, or a sequence segment paired with a label. Each choice determines the query unit, admissible history or context, entity grouping, and target relation. We formalize these choices with a unified prediction interface that separates the query-time information set from the predictor applied to it. The interface is a descriptive protocol language, not a formal reduction theory; its value is to make evidence packaging explicit before comparing model families.

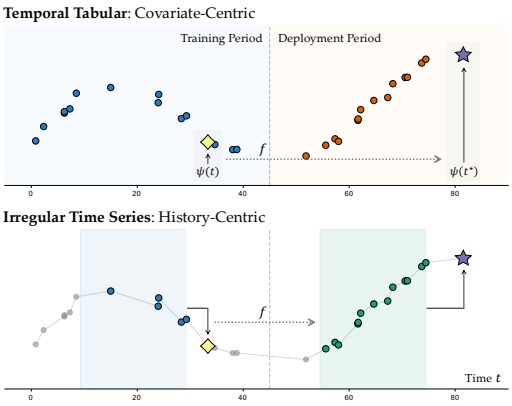


Figure 1: Two default protocols for temporal prediction. Temporal tabular prediction is commonly *covariate-centric*: the query is a timestamped row and evaluation is on future periods. Irregular time-series prediction is commonly *history-centric*: the query is answered from timestamped observations in an entity history.

This view clarifies two practical failure modes. When temporal tables are ordered only by timestamp, different entities can collapse into one spurious stream. When source sequences are summarized into rows, the relevant temporal relation can move from within a sequence to between unrelated rows. Conversely, fixed-row tabular predictors cannot directly consume variable-length irregular histories. We study both directions of protocol realignment: a context-based tabular predictor applied to irregular tasks through temporal embedding and entity-specific admissible contexts, and two temporal-table cases where recovering entity histories or sequence-to-label inputs makes time-series evaluation well specified.

Our main contributions are as follows:

- We formulate temporal prediction through query-time information sets and evidence-packaging operators, providing a common protocol language for temporal tables and irregular time series.
- We diagnose protocol-induced transfer failures, including entity collapse and temporal relation mismatch, and distinguish them from claims about architectural weakness.
- We instantiate two protocol realignments: a tabular lens that repackages admissible candidate sets as irregular contexts, and two history-recovery case studies for selected temporal tables.

## 2 Related Work

We review *irregular time-series prediction* and *tabular prediction under temporal shift*, two settings often derived from similar data sources but studied in separate communities.

### 2.1 Irregular Time-Series Prediction

Irregular time-series prediction addresses uneven sampling, missing values, and asynchronous measurements (Box et al., 2015; Moebus et al., 2025). In conventional forecasting, autoregressive models such as DeepAR (Salinas et al., 2020) build on recurrent architectures such as LSTM/GRU (Hochreiter & Schmidhuber, 1997; Chung et al., 2014), while direct multi-step methods predict a future segment jointly (Zeng et al., 2023a; Nie et al., 2023; Han et al., 2024).

A central question is how to represent a *history window* with non-uniform timestamps. Recurrent and filtering approaches (Che et al., 2018; Becker et al., 2019; Brouwer et al., 2019) maintain a hidden state, with time-aware variants accounting for missingness and irregular intervals. Continuous-time latent models (Chen et al., 2018; Rubanova et al., 2019; Zeng et al., 2023b) evolve states between observations using differential equations. Neural Flows (Bilos et al., 2021) model solution curves as an efficient alternative, while CRU (Schirmer et al., 2022) models a continuous latent state with a linear SDE. Attention-based interpolation offers another approach: mTAN (Shukla & Marlin, 2021) constructs query-time representations by attending over irregular observations with continuous-time embeddings. ACSSM (Park et al., 2025) uses Transformer-based assimilation for simulation-free

Table 1: Comparison of default temporal prediction protocols. A checkmark indicates that the default protocol explicitly exposes the structure to the learner; (✓) indicates that it is required by the standard formulation; × means that the structure is not explicitly exposed, even if it may exist in the raw data.

| **Settings** | Multi-entity | Multi-variate | Covariates | Look-back | Irregularity | Inter-polation | Overlap |
|---|---|---|---|---|---|---|---|
| Regular Time Series | ✓ | ✓ | ✓ | (✓) | × | × | × |
| Irregular Time Series | ✓ | ✓ | ✓ | (✓) | ✓ | ✓ | × |
| *I.i.d*. Tabular | × | × | (✓) | × | × | (✓) | (✓) |
| Temporal Tabular | × | × | (✓) | × | ✓ | ✓ | ✓ |
| **Representable by our interface** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

inference. Temporal point processes provide another event-sequence view of irregular observations, modeling event intensities rather than fixed targets at query times (Du et al., 2016; Mei & Eisner, 2017). They fit our taxonomy as protocols whose $\mathcal{I}_{\text{back}}$ is an event history and whose target is an event-time or mark distribution, but they are not the main empirical focus of this paper.

### 2.2 Tabular Prediction under Temporal Shift

Temporal tabular prediction studies supervised learning on time-stamped, covariate-rich records with the goal of generalizing from past periods to future ones under temporal distribution shift (Rubachev et al., 2025). Although much of tabular learning is developed under an *i.i.d*. assumption (Jiang et al., 2025), deployment is often non-*i.i.d*.: feature distributions drift, population composition changes, and feature semantics may evolve (Zhou, 2022; Gardner et al., 2023; Cai & Ye, 2025a). Strong tabular predictors therefore remain central, including tree-based methods (Breiman, 2001; Chen & Guestrin, 2016; Ke et al., 2017; Prokhorenkova et al., 2018) and deep tabular models that improve representation learning and interaction modeling (Gorishniy et al., 2021; 2022; 2024; 2025; Ye et al., 2025; Holzmüller et al., 2024; Hollmann et al., 2023; 2025; Liu & Ye, 2025; Qu et al., 2025).

Related methods address distribution shift via domain adaptation and covariate-shift formulations (Sugiyama et al., 2007; Ganin et al., 2016; Lipton et al., 2018; Sun & Saenko, 2016; Kim et al., 2024), but these often assume relatively static source-to-target transfer and access to target-domain data ahead of time, which does not fully capture continuously evolving temporal shift (Zhou, 2022; Cai et al., 2024). Recent steps toward temporal adaptivity inject time information into tabular models, *e.g*., temporal embeddings to capture trends and periodicities (Cai & Ye, 2025a;b).

### 2.3 Connection and Separation

The two settings can be derived from the same longitudinal records or event logs; the difference is often in how available evidence is packaged and how evaluation is framed. Irregular time-series methods emphasize time-indexed evidence and explicit temporal structure under irregular sampling (Che et al., 2018; Rubanova et al., 2019; Park et al., 2025), whereas temporal tabular methods emphasize covariate modeling and adaptation to temporal shift (Cai & Ye, 2025a;b). Their largely separate development has fragmented benchmarks and limited cross-pollination, motivating the unified prediction interface proposed in this work.

## 3 Preliminaries

### 3.1 A Unified Temporal Prediction Interface

Let $q$ denote a prediction query, which may include an entity, a timestamp, a target channel, or a prediction horizon. A temporal prediction *protocol* specifies an evidence-packaging operator $\mathcal{A}_{\mathcal{P}}$ that maps the available data $\mathcal{D}$ and query $q$ to the information set

$$\mathcal{I}_{\mathcal{P}}(q) = \mathcal{A}_{\mathcal{P}}(\mathcal{D}, q). \tag{1}$$

The predictor then answers

$$\hat{y}(q) \;=\; f\big(\mathcal{I}_{\mathcal{P}}(q),\, q\big), \tag{2}$$

where $f$ may use time, entity information, or horizon information encoded in $q$. Given training examples $\mathcal{D}_{\text{train}} = \{(\mathcal{I}_{\mathcal{P}}(q), q, y(q))\}$, we learn $f$ by empirical risk minimization (Vapnik, 1991),

$$f^\star \in \arg\min_f \sum_{(\mathcal{I}_{\mathcal{P}}(q),q,y(q))\in\mathcal{D}_{\text{train}}} \mathcal{L}\big(f(\mathcal{I}_{\mathcal{P}}(q), q),\, y(q)\big)\,, \tag{3}$$

with a task-dependent loss $\mathcal{L}$; we write $q = t$ when the timestamp alone identifies the query. To connect common task specifications, we decompose the information set into a *history/context* component and an *exogenous/known-ahead* component:

$$\mathcal{I}(t) \triangleq \big(\mathcal{I}_{\text{back}}(t),\, \mathcal{I}_{\text{fore}}(t)\big), \tag{4}$$

where $\mathcal{I}_{\text{back}}(t)$ contains admissible observed evidence (*e.g.*, a causal look-back window or an interpolation context) and $\mathcal{I}_{\text{fore}}(t)$ contains covariates known at query time, including any available over the forecast horizon. Either part may be empty depending on the protocol.

**Irregular time-series prediction**. For an entity $u$, irregular observations arrive at non-uniform timestamps. An extrapolation protocol constructs a causal history

$$\mathcal{I}_{\text{back}}(u,t) = \{(t_{u,i},\, y_{u,i},\, \boldsymbol{x}_{u,i}) : t - L < t_{u,i} < t\}, \tag{5}$$

where $\boldsymbol{x}_{u,i}$ denotes optional covariates observed with $y_{u,i}$. Future known covariates, when available, form

$$\mathcal{I}_{\text{fore}}(u,t) = \{(t_q,\, \boldsymbol{x}_{u,q}) : t \le t_q < t + H\}. \tag{6}$$

In interpolation, the admissible context is not necessarily causal: it consists of observed, unmasked measurements surrounding the masked query time, with the queried value itself excluded. Thus both regimes are specified by $\mathcal{A}_{\mathcal{P}}$, but they differ in which observations are admissible.

The core challenge is to extract predictive structure from sparse and irregular evidence. Irregular sampling requires models to reason about time gaps; *asynchrony* further allows different variables/channels to be recorded at different timestamps, and *overlap* can occur when observation times coincide with query times. In many pipelines, observations are represented as event sequences; when multiple records share the same timestamp, preprocessing such as aggregation or tie-breaking is often required before feeding the data to sequential models.

**Tabular prediction under temporal shift**. Temporal tabular prediction treats each time-stamped record as a covariate-rich example $(t, \boldsymbol{x}, y)$. The default specification uses no explicit temporal context window and takes the information at query time to be the current covariates (and time, if provided),

$$\mathcal{I}_{\text{back}}(t) = \emptyset, \qquad \mathcal{I}_{\text{fore}}(t) = (t,\, \boldsymbol{x}), \tag{7}$$

with targets $y \in \mathbb{R}$ (regression) or $y \in \{1, \ldots, C\}$ / $\mathbb{R}^C$ (classification). Generalization is defined by *time-based evaluation*: the model is trained on earlier periods and tested on later periods, so success means robust performance under temporal distribution shift.

The key challenge is that the mapping from heterogeneous covariates to targets can evolve over time due to changing populations, seasonality, and interventions; feature semantics may drift as measurements change meaning or availability. When the underlying data are collected asynchronously at finer temporal granularity than the instance timestamp, tabular pipelines typically absorb asynchrony into missing values or engineered summaries rather than preserving event-level timing.

**Special cases and a compact taxonomy**. Regular time series is the special case where timestamps lie on a fixed grid and history/horizon are indexed by discrete steps:

$$\mathcal{I}_{\text{back}}(t) = \{(y_i, \boldsymbol{x}_i)\}_{i=t-L}^{t-1}, \quad \mathcal{I}_{\text{fore}}(t) = \{\boldsymbol{x}_i\}_{i=t}^{t+H-1}. \tag{8}$$

*I.i.d.* tabular learning is also a special case of Equation (7) without time, where $\mathcal{I}_{\text{fore}}(t) = \boldsymbol{x}$ and evaluation follows the usual *i.i.d.* test protocol. We summarize these common settings and their assumptions in Table 1.

### 3.2 Protocol Changes and Compatibility

The interface is useful only when it records what a protocol changes. Temporal embedding changes the query representation by adding a learned representation of time; temporal modulation changes

Table 2: Comparison of irregular time series methods on temporal tabular benchmark (Rubachev et al., 2025). The reported metric is performance percentage change, *i.e.*, a positive percentage change indicates that the method outperforms static MLP.

| Settings | Methods | Avg. Imp. |
|---|---|---|
| *I.i.d.* Tabular | MLP (Gorishniy et al., 2022) | 0.00% |
| Irregular Time Series | GRU-D (Che et al., 2018) | −5.06% |
| | CRU (Schirmer et al., 2022) | −8.41% |
| | ACSSM (Park et al., 2025) | −5.12% |
| | Neural Flows (Bilos et al., 2021) | −3.35% |
| | mTAND (Shukla & Marlin, 2021) | −35.14% |
| Temporal Tabular | MLP (+e) (Cai & Ye, 2025b) | +1.31% |
| | MLP (+m) (Cai & Ye, 2025a) | +2.09% |

how time affects feature representations; context construction changes $\mathcal{A}_{\mathcal{P}}$ by restricting which labeled examples are admissible; entity recovery changes the grouping of events; and sequence-to-label recasting changes the atomic query unit. We assess protocol compatibility by whether $\mathcal{I}_{\mathcal{P}}(q)$ exposes the evidence expected by the model class. For sequence or continuous-time models, ordered events should correspond to a coherent entity or source sequence. For fixed-row tabular models, variable-length histories must be represented through admissible context or summarized features. This is the sense in which Table 1 compares protocols: a cross indicates that the default protocol does not explicitly expose the corresponding temporal structure, not that the raw data can never contain it.

## 4 The Gap between Task Specifications

The interface in Section 3 makes protocols comparable, but it does not make arbitrary format conversions valid. We examine whether $\mathcal{A}_{\mathcal{P}}$ supplies evidence consistent with the model's input assumptions. We first use direct conversions as stress tests, then study recoveries in Sections 5 and 6.

We use the following notation for protocol variants in the empirical tables. (+e) denotes *temporal embedding*: the model receives a learned representation of the query time. (+m) denotes *temporal modulation*: time is used to modulate feature representations. (+c) denotes *context construction*: the candidate or support set is restricted to protocol-admissible temporal evidence. (+ec) combines temporal embedding and context construction.

### 4.1 Irregular Models on Temporal Tables

We first apply irregular time-series architectures directly to temporal tabular benchmarks. Timestamped rows are ordered by time and fed as events, with row covariates treated as observed features. Table 2 summarizes the average performance change on TabReD (Rubachev et al., 2025). On average, the direct irregular time-series adaptations underperform a static MLP baseline, whereas MLP (+e) and MLP (+m) improve over it.

This comparison tests the direct conversion, not the general capability of these architectures. Temporal tables often contain many records at the same coarse timestamp, and those records may correspond to different entities or independent events. A recurrent or continuous-time model expects ordered events to define a coherent state process. Under global row ordering, the model must aggregate concurrent rows, impose an arbitrary within-time order, or predict from nearly unchanged latent states. The resulting transitions need not correspond to the temporal dynamics of any entity.

If the table can be decomposed into meaningful entity histories, or if each row can be expanded back into its underlying event sequence, then a time-series model may again receive the evidence it expects. Section 6 studies exactly these recoveries.

### 4.2 Tabular Models on Irregular Histories

The reverse transfer has a different obstacle. A standard tabular predictor expects a fixed-dimensional query row and may additionally use query time. An irregular time-series query instead exposes a variable-length admissible context: different entities may have different numbers of observations, different observation times, and different missingness patterns within the same window.

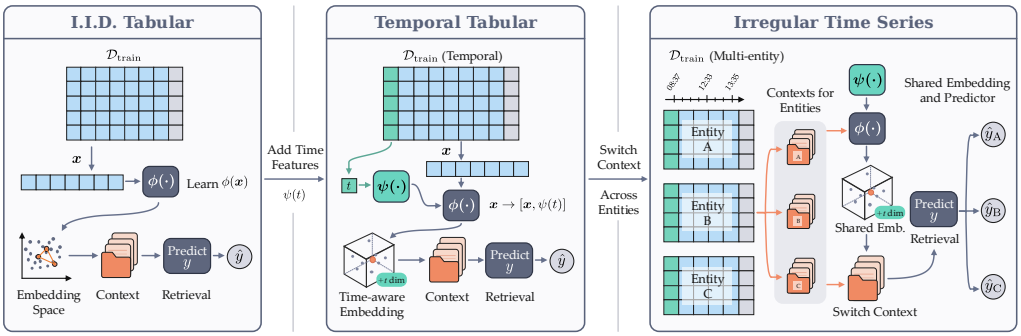


Figure 2: A tabular lens on irregular time series. Temporal embedding changes the query representation, while context construction changes the admissible evidence: the global candidate pool is repackaged into an entity-specific context defined by the irregular time-series protocol.

Flattening such histories into fixed-dimensional features is possible, but it changes the protocol. Padding requires a fixed-size representation, while aggregation and hand-crafted summaries can discard event timing and local evidence. Therefore, a fixed-row tabular learner cannot directly consume the information set $\mathcal{I}_{\text{back}}(t)$ in Equation (5). We therefore repackage the admissible context to preserve variable-length temporal evidence. Section 5 shows how context-based tabular predictors provide such a route by treating the admissible history as the query-specific context.

## 5 A Tabular Lens on Irregular Time Series

This section studies whether a context-based tabular predictor can be evaluated under irregular time-series protocols once $\mathcal{A}_{\mathcal{P}}$ exposes the proper evidence. As shown in Figure 2, the required changes are protocol-level: *temporal embedding* represents query time, and *context construction* replaces the global candidate pool with an admissible entity-specific context.

### 5.1 Temporal Embedding and Context Construction

Context-based tabular methods, including retrieval-based predictors such as TabR (Gorishniy et al., 2024) and ModernNCA (Ye et al., 2025) and in-context predictors such as TabPFN (Hollmann et al., 2023; 2025), answer a query using a labeled support or candidate set. Let $r(t)$ denote the representation of query $q$ at time $t$, and let $\mathcal{C}(t) = \{(r_j, y_j)\}_{j=1}^{n_t}$ denote the labeled context admitted by the protocol. A simplified retrieval-style instance embeds the query and context elements, computes similarity weights, and aggregates labels:

$$w_j(t) = \frac{\exp\big(-d(h_\theta(r(t)), h_\theta(r_j))/\tau\big)}{\sum_{k=1}^{n_t} \exp\big(-d(h_\theta(r(t)), h_\theta(r_k))/\tau\big)}, \quad \hat{y}(t) = \sum_{j=1}^{n_t} w_j(t) y_j, \tag{9}$$

where $h_\theta$ is a learned representation, $d(\cdot,\cdot)$ is a distance, and $\tau > 0$ is a temperature. PFN-style and learned-readout variants replace the aggregation rule, but the protocol-level dependence on an admissible context remains. We distinguish a global training support set used by $f$ from query-specific temporal evidence in $\mathcal{I}_{\mathcal{P}}(q)$.

**From *i.i.d.* tabular to temporal tabular.** In *i.i.d.* tabular prediction, the query representation is $r = \boldsymbol{x}$, and the context is usually a global labeled training pool. Temporal tabular prediction changes the query protocol by attaching time to each row. Prior work implements this by learning a temporal embedding $\psi(t)$ and forming a time-conditioned query

$$r(t) = [\boldsymbol{x}_t, \psi(t)]. \tag{10}$$

Under our interface, this modifies the query representation used by $f$ without changing the admissible evidence: $\mathcal{I}_{\text{back}}(t)$ remains empty, while the model can adapt the covariate–target relationship across calendar time. This is the (+e) protocol variant introduced in Section 4 and corresponds to the middle panel of Figure 2.

Table 3: **Main results on the irregular time series datasets** Physionet (Silva et al., 2012), USHCN (Menne et al., 2015), and Human Activity (Cook et al., 2019). The best performance is shown in **bold**, with the 2nd best result underlined. Physionet and USHCN use RMSE↓, while Human Activity uses AUC↑.

| Methods | Interpolation | | Extrapolation | | Classification | Avg. Rank |
|---|---|---|---|---|---|---|
| | Physionet↓ | USHCN↓ | Physionet↓ | USHCN↓ | Human Act.↑ | |
| *Irregular Time Series* | | | | | | |
| GRU-D (Che et al., 2018) | 0.0968 | 0.0984 | 0.0942 | 0.1305 | – | 6.25 |
| CRU (Schirmer et al., 2022) | 0.1068 | 0.0968 | 0.0997 | 0.1142 | – | 5.75 |
| ACSSM (Park et al., 2025) | 0.1075 | 0.0961 | 0.1046 | 0.1287 | 0.9533 | 6.20 |
| Latent ODE (Rubanova et al., 2019) | 0.1114 | 0.1266 | 0.0920 | 0.1283 | 0.9216 | 7.00 |
| Neural Flows (Bilos et al., 2021) | 0.1043 | 0.1274 | 0.1029 | 0.1293 | 0.9403 | 7.60 |
| mTAND (Shukla & Marlin, 2021) | 0.0810 | 0.1009 | 0.0930 | 0.1193 | – | 5.00 |
| HyperIMTS (Li et al., 2025) | 0.0688 | 0.1053 | 0.0686 | 0.1101 | 0.8362 | 3.60 |
| GraFITi (Yalavarthi et al., 2024) | 0.0694 | 0.1076 | **0.0680** | 0.1107 | – | 3.75 |
| Hi-Patch (Luo et al., 2025) | **0.0658** | 0.1061 | 0.0710 | 0.1112 | **0.9719** | **3.20** |
| *Context-based Tabular* | | | | | | |
| ModernNCA (+ec) | 0.0987 | **0.0948** | 0.1018 | **0.1072** | 0.9234 | 4.00 |

**From temporal tabular to irregular time series.** Irregular time-series prediction further changes $\mathcal{A}_{\mathcal{P}}$. The useful evidence is no longer a global pool of unrelated rows; it is the context admitted for the queried entity and task. Let $u$ index entities such as patients, stations, or devices, and let the query be $(u, t)$. We instantiate the context as

$$\mathcal{C}_u^{\mathcal{P}}(t) \;=\; \{(r_{u,j}, y_{u,j}) : (t_{u,j}, y_{u,j}, \boldsymbol{x}_{u,j}) \in \mathcal{I}_{\text{back}}(u, t)\}, \tag{11}$$

where $r_{u,j} = [\boldsymbol{x}_{u,j}, \psi(t_{u,j})]$. For extrapolation, $\mathcal{A}_{\mathcal{P}}$ admits causal observations before $t$; for interpolation, it admits observed unmasked measurements around the query time while excluding the masked target. We use $\mathcal{C}_u^{\mathcal{P}}(t)$ as the support set in Equation (9), with the query represented as in Equation (10). This recovers the history/context component of $\mathcal{I}_{\mathcal{P}}(q)$ without forcing a variable-length window into a fixed-dimensional row. It also avoids imposing a recurrent or continuous-time state evolution on concurrent observations: the history is a set of admissible evidence, not a single latent trajectory that must advance through every timestamp.

### 5.2 Evaluation on Irregular Time Series

We instantiate the protocol with ModernNCA because it is a compact retrieval model whose candidate set can be changed without altering the backbone. The resulting protocol adaptation, denoted ModernNCA (+ec), combines temporal embedding (+e) and entity-specific context construction (+c), and is evaluated on standard irregular time-series benchmarks under interpolation, extrapolation, and classification. Each benchmark instance provides an entity-indexed event history, which directly instantiates $\mathcal{I}_{\text{back}}(t)$ as in Equation (11), while query times and any known covariates form $\mathcal{I}_{\text{fore}}(t)$. Unlike sequence models that summarize the entire history through a recurrent state or continuous-time latent dynamics, ModernNCA (+ec) answers each query by retrieving relevant labeled evidence from the entity-specific context.

For this instantiation, removing either component changes the intended task interface. Without temporal embedding, retrieval has no explicit representation of observation times. Without context construction, the shared candidate pool mixes entities; the unmodified interface would instead require a separate model per entity to keep these pools separate. We therefore evaluate the complete protocol adaptation.

Table 3 shows that this instantiation is competitive on some irregular benchmarks, especially USHCN, but does not uniformly dominate specialized time-series architectures. These results support the use of a tabular retrieval model with entity-specific temporal contexts. Additional PFN-style and running-time results are given in Section B.

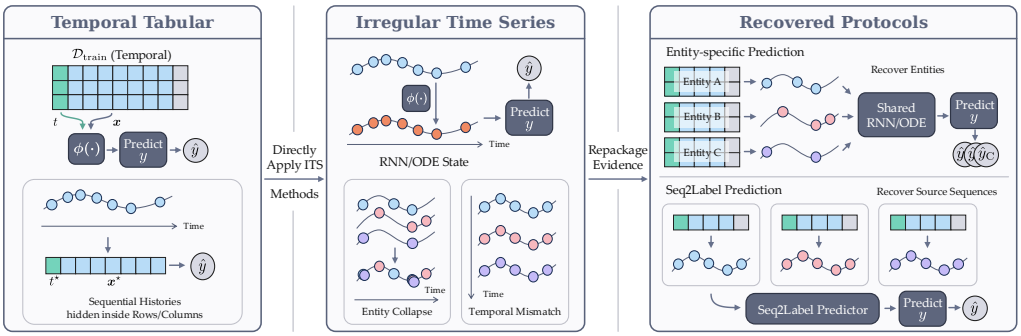


Figure 3: Recovering time-series protocols from temporal tables. Directly applying an irregular time-series model to a temporal table can create spurious sequences: entity collapse interleaves records from different entities, while temporal relation mismatch asks the model to compare across rows even when the predictive signal lies within each row's underlying sequence. Protocol recovery restores the entity-specific or sequence-to-label structure before applying temporal architectures.

## 6 Recovering Histories from Temporal Tables

We now study the reverse direction: applying time-series models to temporal tables whose raw data retain entity histories or source sequences. Figure 3 illustrates two protocol recoveries that address the direct-transfer failures in Section 4.

### 6.1 Entity Collapse and Entity-Resolved Histories

*Entity collapse* occurs when global time ordering interleaves records from different entities, while entity identifiers remain ordinary covariates. A time-series model then evolves its state through transitions such as

$$(u_i, t_i, \boldsymbol{x}_i, y_i) \to (u_j, t_j, \boldsymbol{x}_j, y_j), \qquad u_i \neq u_j,$$

which need not represent the temporal evolution of either entity. Concurrent records may therefore reflect distinct entity histories rather than repeated observations within one sequence.

We recover *entity-resolved histories* by defining the evidence for a query $(u, t)$ as

$$\mathcal{H}_u(t) = \{(t_{u,j}, y_{u,j}, \boldsymbol{x}_{u,j}) : t_{u,j} < t\}. \tag{12}$$

Setting $\mathcal{I}_{\text{back}}(u, t) = \mathcal{H}_u(t)$ replaces the global stream with same-entity evidence without changing the temporal architecture.

**Weather: separating station histories.** In TabReD's Weather (WE) dataset, each row is a station observation; station identity, latitude, and longitude are treated as covariates. The direct conversion orders all rows by timestamp, creating a single sequence across stations. Although station identity is available as a feature, the temporal state still evolves across station boundaries.

We partition the table by station and construct a separate history for each station. For a query $(u, t)$, the model receives only earlier observations from station $u$, with their timestamps and covariates, as specified by Equation (12). The prediction target and model architecture remain unchanged. Table 4 shows improvements for four of five methods on WE, although all remain behind MLP (+m). Entity recovery thus helps these models use station histories, but does not eliminate the advantage of the tabular baseline. The response also varies by architecture: mTAND benefits substantially from separating stations, whereas CRU changes little and slightly worsens. This variation suggests that coherent entity histories are useful inputs, but their value still depends on how the predictor models the available evidence.

### 6.2 Temporal Relation Mismatch and Sequence-to-Label Recasting

*Temporal relation mismatch* arises when each row summarizes an underlying sequence through statistics, counts, or other descriptors. Ordering these rows as a sequence asks a time-series model to learn transitions *between* summarized instances, while potentially predictive temporal structure lies *within* each source sequence.

Table 4: Protocol recovery on EO (AUC↑) and WE (RMSE↓). Δ is Reframed–Default; positive is better for EO and negative is better for WE.

| Data | Method | Def. | Refr. | Δ | MLP | +e | +m |
|---|---|---|---|---|---|---|---|
| EO | GRU-D | 0.5732 | 0.5822 | +0.0090 | 0.5866 | 0.5877 | 0.5832 |
| | ACSSM | 0.5631 | 0.5869 | +0.0238 | | | |
| | mTAND | 0.4993 | 0.5885 | +0.0892 | | | |
| | CRU | 0.5148 | 0.5731 | +0.0583 | | | |
| | Neural Flows | 0.5779 | **0.5914** | +0.0135 | | | |
| WE | GRU-D | 1.6533 | **1.5941** | -0.0592 | 1.5331 | 1.5363 | 1.5157 |
| | ACSSM | 1.6286 | 1.5987 | -0.0299 | | | |
| | mTAND | 2.3823 | 1.6274 | -0.7549 | | | |
| | CRU | 1.6032 | 1.6045 | +0.0013 | | | |
| | Neural Flows | 1.6131 | 1.6113 | -0.0018 | | | |

We recast the task as *sequence-to-label prediction* by recovering the source sequence for instance $i$:

$$\mathcal{Q}_i = \{(\boldsymbol{z}_{i,k}, \tau_{i,k})\}_{k=1}^{m_i}, \tag{13}$$

where $\boldsymbol{z}_{i,k}$ and $\tau_{i,k}$ are the event features and times within the segment. For query $q = i$, we set $\mathcal{I}_{\text{back}}(i) = \mathcal{Q}_i$ and $\mathcal{I}_{\text{fore}}(i) = t_i$, where $t_i$ is the instance timestamp. The predictor in Equation (2) maps this variable-length sequence and timestamp to the instance label. Recovery requires access to the underlying event records; the statistical summaries alone generally do not determine the source sequence.

**Ecom-Offers: recovering transaction sequences.** Ecom-Offers (EO) predicts whether a user redeems a coupon after receiving an offer. The tabular benchmark summarizes the user's variable-length transaction history into fixed-dimensional features and timestamps each row by the coupon issue time. These summaries can encode user behavior and market conditions, but ordering coupon rows by issue time does not expose the within-history event structure to a time-series model.

For each coupon instance $i$, we construct $\mathcal{Q}_i$ from the user's transactions strictly before the offer. Each event retains transaction-derived features and its time relative to the offer; the target remains coupon redemption. The model learns an irregular sequence-to-label classification task over these purchase histories. Table 4 shows that recasting improves all five methods on EO; Neural Flows and mTAND achieve higher mean AUC than the strongest MLP baseline.

The two recoveries modify different aspects of the interface. WE retains the station-time query and changes which observations may enter its history; EO retains the coupon-level target but replaces summary features with the source transaction sequence. In both cases, retaining the original target makes the comparison relevant to the tabular task, while the recovered inputs expose temporal relations appropriate for a sequence model. The gains therefore concern these concrete changes to the available evidence, rather than a general ranking of tabular and time-series architectures. Our analysis covers only WE and EO, whose raw data retain station histories and transaction sequences; its applicability to other tabulation mechanisms remains to be tested.

## 7 Conclusion

We presented a protocol-level interface for temporal prediction that makes query-time evidence explicit. Rather than treating temporal tabular prediction and irregular time-series prediction as separated purely by modality, the interface describes how a protocol packages evidence as rows, contexts, entity histories, or sequence-to-label examples. This framing identifies protocol mismatches that can impair direct transfer and confound model comparisons. Empirically, context-based tabular prediction becomes applicable to irregular tasks when candidate sets are repackaged as temporally embedded, entity-specific contexts, while selected temporal tables support time-series evaluation after entity histories or source sequences are recovered. Repackaging provides a practical basis for cross-domain reuse: specify the prediction unit and admissible evidence, then assess which model can use that evidence effectively. Future benchmarks should document entity boundaries, source histories, and query-time availability alongside model performance, making protocol differences explicit and supporting reproducible comparisons across temporal prediction settings.

### AI USE STATEMENT

We used generative AI tools to polish the manuscript and provide editorial feedback on the presentation of protocols and interpretation of results. The authors reviewed the revisions and take full responsibility for the final content.

### ETHICS STATEMENT

This work uses established public benchmarks and involves no new data collection or human-subject experiments. We credit the datasets and methods and follow the licenses and access conditions of the data and software used. Downstream risks include unreliable predictions under distribution shift and privacy leakage from historical context. Applications should use domain-specific validation, data minimization, and appropriate access controls.

### REPRODUCIBILITY STATEMENT

The prediction interface and protocol transformations are specified in Sections 3 and 4. Sections 5 and 6 describe the empirical comparisons. Appendix A documents the software and hardware environment, data splits, evaluation metrics, training procedure, tuning budget, and model-specific hyperparameter search spaces; Appendix B provides additional protocol details and baseline results. Reported performance is averaged over 15 random seeds. The current submission does not include an accompanying code release with exact reproduction commands.

## REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *KDD*, pp. 2623–2631, 2019.

Philipp Becker, Harit Pandya, Gregor H. W. Gebhardt, Cheng Zhao, C. James Taylor, and Gerhard Neumann. Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces. In *ICML*, pp. 544–552, 2019.

Marin Bilos, Johanna Sommer, Syama Sundar Rangapuram, Tim Januschowski, and Stephan Günnemann. Neural flows: Efficient alternative to neural odes. In *NeurIPS*, pp. 21325–21337, 2021.

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

Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. In *NeurIPS*, pp. 7377–7388, 2019.

Hao-Run Cai and Han-Jia Ye. Feature-aware modulation for learning from temporal tabular data. *CoRR*, abs/2512.03678, 2025a.

Hao-Run Cai and Han-Jia Ye. Understanding the limits of deep tabular methods with temporal shift. In *ICML*, pp. 6366–6386, 2025b.

Zekun Cai, Guangji Bai, Renhe Jiang, Xuan Song, and Liang Zhao. Continuous temporal domain generalization. In *NeurIPS*, pp. 127987–128014, 2024.

Ching Chang, Jeehyun Hwang, Yidan Shi, Haixin Wang, Wen-Chih Peng, Tien-Fu Chen, and Wei Wang. Time-imm: A dataset and benchmark for irregular multimodal multivariate time series. *CoRR*, abs/2506.10412, 2025.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8:6085, 2018.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *NeurIPS*, pp. 6572–6583, 2018.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *KDD*, pp. 785–794, 2016.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

Diane J. Cook, Aaron S. Crandall, and Brian L. Thomas. Human activity recognition from continuous ambient sensor data. UCI Machine Learning Repository, 2019.

Nan Du, Hanjun Dai, Rakshit S. Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, pp. 1555–1564, 2016.

Deepak Ganesan, Sylvia Ratnasamy, Hanbiao Wang, and Deborah Estrin. Coping with irregular spatio-temporal sampling in sensor networks. *Comput. Commun. Rev.*, 34(1):125–130, 2004.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35, 2016.

Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Benchmarking distribution shift in tabular data with tableshift. In *NeurIPS*, pp. 53385–53432, 2023.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *NeurIPS*, pp. 18932–18943, 2021.

Yury Gorishniy, Ivan Rubachev, and Artem Babenko. On embeddings for numerical features in tabular deep learning. In *NeurIPS*, pp. 24991–25004, 2022.

Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. Tabr: Tabular deep learning meets nearest neighbors. In *ICLR*, 2024.

Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. Tabm: Advancing tabular deep learning with parameter-efficient ensembling. In *ICLR*, 2025.

Lu Han, Han-Jia Ye, and De-Chuan Zhan. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *IEEE Trans. Knowl. Data Eng.*, 36:7129–7142, 2024.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *ICLR*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025.

David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned mlps and boosted trees on tabular data. In *NeurIPS*, pp. 26577–26658, 2024.

Shi Bin Hoo, Samuel Muller, David Salinas, and Frank Hutter. From tables to time: Extending tabpfn-v2 to time series forecasting. *CoRR*, abs/2501.02945, 2025.

Jun-Peng Jiang, Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, and Han-Jia Ye. Representation learning for tabular data: A comprehensive survey. *CoRR*, abs/2504.16109, 2025.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *NIPS*, pp. 3146–3154, 2017.

Changhun Kim, Taewon Kim, Seungyeon Woo, June Yong Yang, and Eunho Yang. Adaptable: Test-time adaptation for tabular data via shift-aware uncertainty calibrator and label distribution handler. *CoRR*, abs/2407.10784, 2024.

Boyuan Li, Yicheng Luo, Zhen Liu, Junhao Zheng, Jianming Lv, and Qianli Ma. HyperIMTS: Hypergraph neural network for irregular multivariate time series forecasting. In *ICML*, pp. 35502–35518, 2025.

Zachary C. Lipton, Yu-Xiang Wang, and Alexander J. Smola. Detecting and correcting for label shift with black box predictors. In *ICML*, pp. 3128–3136, 2018.

Si-Yang Liu and Han-Jia Ye. Tabpfn unleashed: A scalable and effective solution to tabular classification problems. In *ICML*, pp. 40043–40068, 2025.

Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, Huai-Hong Yin, Tao Zhou, Jun-Peng Jiang, and Han-Jia Ye. TALENT: A tabular analytics and learning toolbox. *J. Mach. Learn. Res.*, 26:226:1–226:16, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Yicheng Luo, Bowen Zhang, Zhen Liu, and Qianli Ma. Hi-patch: Hierarchical patch GNN for irregular multivariate time series. In *ICML*, pp. 41494–41519, 2025.

Hongyuan Mei and Jason Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *NIPS*, pp. 6754–6764, 2017.

MJ Menne, CN Williams Jr, RS Vose, and Data Files. Long-term daily climate records from stations across the contiguous united states, 2015.

Max Moebus, Björn Braun, and Christian Holz. Contimask: Explaining irregular time series via perturbations in continuous time. In *NeurIPS*, 2025.

Lennart J. Nederstigt, Steven S. Aanen, Damir Vandic, and Flavius Frasincar. FLOPPIES: A framework for large-scale ontology population of product information from tabular data in e-commerce stores. *Decis. Support Syst.*, 59:296–311, 2014.

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

Byoungwoo Park, Hyungi Lee, and Juho Lee. Amortized control of continuous state space feynman-kac model for irregular time series. In *ICLR*, 2025.

Liudmila Ostroumova Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *NeurIPS*, pp. 6639–6649, 2018.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. In *ICML*, pp. 50817–50847, 2025.

Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M. Dai, Nissan Hajaj, Michaela Hardt, Peter J. Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, Patrik Sundberg, Hector Yee, Kun Zhang, Yi Zhang, Gerardo Flores, Gavin E. Duggan, Jamie Irvine, Quoc Le, Kurt Litsch, Alexander Mossin, Justin Tansuwan, De Wang, James Wexler, Jimbo Wilson, Dana Ludwig, Samuel L. Volchenboum, Katherine Chou, Michael Pearson, Srinivasan Madabushi, Nigam H. Shah, Atul J. Butte, Michael D. Howell, Claire Cui, Gregory S. Corrado, and Jeffrey Dean. Scalable and accurate deep learning with electronic health records. *npj Digit. Medicine*, 1, 2018.

Ivan Rubachev, Nikolay Kartashev, Yury Gorishniy, and Artem Babenko. Tabred: A benchmark of tabular machine learning in-the-wild. In *ICLR*, 2025.

Yulia Rubanova, Tian Qi Chen, and David Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *NeurIPS*, pp. 5321–5331, 2019.

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.

Mona Schirmer, Mazin Eltayeb, Stefan Lessmann, and Maja Rudolph. Modeling irregular time series with continuous recurrent units. In *ICML*, pp. 19388–19405, 2022.

Satya Narayan Shukla and Benjamin M. Marlin. Multi-time attention networks for irregularly sampled time series. In *ICLR*, 2021.

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. In *2012 computing in cardiology*, pp. 245–248, 2012.

Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. Covariate shift adaptation by importance weighted cross validation. *J. Mach. Learn. Res.*, 8:985–1005, 2007.

Baochen Sun and Kate Saenko. Deep CORAL: correlation alignment for deep domain adaptation. In *ECCV Workshops (3)*, pp. 443–450, 2016.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *NIPS*, pp. 831–838, 1991.

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 *AAAI*, pp. 16255–16263, 2024.

Han-Jia Ye, Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, and De-Chuan Zhan. A closer look at deep learning methods on tabular datasets. *CoRR*, abs/2407.00956, 2024.

Han-Jia Ye, Huai-Hong Yin, De-Chuan Zhan, and Wei-Lun Chao. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. In *ICLR*, 2025.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, pp. 11121–11128, 2023a.

Sebastian Zeng, Florian Graf, and Roland Kwitt. Latent sdes on homogeneous spaces. In *NeurIPS*, pp. 76151–76180, 2023b.

Zhi-Hua Zhou. Open environment machine learning. *National Science Review*, 9:nwac123, 2022.

# A Additional Experimental Setup

## A.1 Training

All experiments are conducted on Linux using Python 3.11 and PyTorch 2.7.0. Experiments are run on machines equipped with NVIDIA RTX 4090 GPUs with 24 GB memory and four Intel Xeon Platinum 8352S CPUs. We use a unified batch size for all models on the irregular datasets and on TabReD: mTAND and GRU-D are trained with batch size 1 due to their high computational overhead and large GPU-memory footprint, while all other methods use batch size 128. We optimize all models with AdamW (Loshchilov & Hutter, 2019), apply early stopping with a patience of 16 epochs, and cap training at 200 epochs. Each method is trained with 15 different random seeds, and we report the average performance across runs.

For all models on the irregular datasets and TabReD, we adopt the same task-specific loss and evaluation metric. Following Chang et al. (2025), we first aggregate predictions and labels across all time steps and samples for each channel, and then average the resulting scores over channels; only mask-indicated valid entries are included. For classification, we use masked cross-entropy for training and masked AUC (higher is better) for evaluation. For regression, we use masked MSE for training and masked RMSE (lower is better) for evaluation. On TabReD, the target contains no missing values and there is a single prediction channel, so the masked objectives reduce to their standard (unmasked) counterparts.

## A.2 Data Splitting and Evaluation Protocols

For USHCN (Menne et al., 2015) and PhysioNet (Silva et al., 2012), we follow the splitting protocol of Schirmer et al. (2022): we allocate 80% of sequences to training and 20% to testing, and further hold out 25% of the training split as a validation set for checkpoint selection. All final results are reported on the untouched test set. For TabReD, we use exactly the same preprocessing, training, evaluation, and hyper-parameter tuning pipeline released by Cai & Ye (2025b); Ye et al. (2024); Liu et al. (2025).

We evaluate both interpolation and extrapolation on the same two datasets. **Interpolation:** given a subset of observations $\boldsymbol{x}_{\mathcal{S}}$ with $\mathcal{S} \subseteq \mathcal{T}$, we reconstruct the complete sequence $\boldsymbol{s}_{\mathcal{T}} = \boldsymbol{x}_{\mathcal{T}}$. Prior work (Park et al., 2025; Schirmer et al., 2022; Rubanova et al., 2019) typically reconstructs from the full series (*i.e.*, $\mathcal{S} = \mathcal{T}$); instead, we randomly mask a fixed 20% of time points in each sequence and infer the missing values from the remaining 80%. This setting better reflects practical missingness and provides a fairer comparison across baselines. **Extrapolation:** we split the time axis into $\mathcal{T}_1 = \{t_0, \ldots, t_k\}$ and $\mathcal{T}_2 = \{t_{k+1}, \ldots, t_N\}$, and predict the full sequence on $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$ given only observations from $\mathcal{T}_1$. Following Schirmer et al. (2022), for PhysioNet we use the first 24 h of measurements to forecast the next 24 h; for USHCN we split each series into two equal halves and use the first half to predict the second.

## A.3 Hyperparameter Optimization and Model Modification

We tune hyper-parameters with Optuna (Akiba et al., 2019), running 100 trials per method under a one-day budget. The search space simultaneously covers model behaviour and training cost. Model-specific details are listed below.

**GRU-D** Default hyper-parameters are employed across all tasks, as listed in Table 5. We adopt the GRU-D implementation from Rubanova et al. (2019), extending it with an additional decoder (hidden state size of 32) to generate covariates and other predictive variables. At each RNN step, the model receives the ground-truth input rather than the previous prediction. For interpolation and covariate prediction tasks, the observed time sequence is fed directly into the model to construct the masked observations and covariate channels. For extrapolation, the predicted time series is first zero-filled and then concatenated with the observed sequence to form the complete input. Variable-length sequences are right-padded with zeros for batching purposes.

Table 5: GRU-D hyper-parametes

| Parameter | Distribution |
|---|---|
| `latent_dim` | Fixed 32 |
| `n_units` | Fixed 32 |
| `lr` | Fixed $10^{-4}$ |
| `weight_decay` | Fixed $10^{-5}$ |

**CRU** Hyper-parameters are searched within the predefined option space, as listed in Table 6. We apply exactly the same input construction as GRU-D for interpolation, extrapolation and covariate prediction tasks. Variable-length samples are right-padded with zeros.

Table 6: CRU hyper-parameter search space. `PowerInt[a,b]` denotes integer powers of two in $[2^a, 2^b]$. `Int`$[a, b]$ denotes the set of integers in the closed interval $[a, b]$.

| Parameter | Distribution |
|---|---|
| `cru_lsd` | $\mathrm{PowerInt}[2, 5]$ |
| `cru_hidden_units` | $\mathrm{PowerInt}[2, 5]$ |
| `cru_enc_num_layers` | Fixed 1 |
| `cru_dec_num_layers` | Fixed 1 |
| `cru_num_layers` | $\mathrm{Int}[1, 2]$ |
| `lr` | $\mathrm{LogUniform}[10^{-5}, 10^{-3}]$ |
| `weight_decay` | $\{0, \mathrm{LogUniform}[10^{-6}, 10^{-3}]\}$ |

**ACSSM** Hyper-parameters are searched within option space, as listed in Table 7. We apply exactly the same input construction as GRU-D for interpolation, extrapolation and covariate prediction tasks. Variable-length samples within a batch are right-padded with zeros.

Table 7: ACSSM hyper-parameter search space.

| Parameter | Distribution |
|---|---|
| `state_dim` | $\mathrm{PowerInt}[3, 7]$ |
| `num_basis` | $\mathrm{PowerInt}[3, 5]$ |
| `n_layer` | $\mathrm{Int}[1, 5]$ |
| `lr` | $\mathrm{LogUniform}[10^{-5}, 10^{-3}]$ |
| `weight_decay` | $\{0, \mathrm{LogUniform}[10^{-6}, 10^{-3}]\}$ |

**Latent ODE** Hyper-parameters are searched within option space, as listed in Table 8. Because the model requires a unique global time grid, variable-length samples are handled by aggregating the distinct time points of every sample.

**Neural Flows** Hyper-parameters are searched within the predefined option space, as listed in Table 9. The input consists of observable data, from which three distinct channels can be constructed: covariates, interpolation, and extrapolation. Variable-length samples are right-padded with zeros.

**mTAND** Hyper-parameters are searched within option space, as listed in Table 10. Following Shukla & Marlin (2021), we feed the target time stamps into the final decoder, enabling the model to perform interpolation, extrapolation and covariate tasks. Variable-length samples are right-padded with zeros.

**ModernNCA** The hyper-parameter search space is listed in Table 11, including temporal-embedding orders, the trend option, and `d_embedding`. In the raw PhysioNet 2012 data, timestamps record elapsed hours and minutes since ICU admission, rather than absolute calendar dates (Silva et al., 2012).

Table 8: Latent ODE hyper-parameter search space.

| Parameter | Distribution |
|---|---|
| latents | PowerInt$[2, 5]$ |
| units | PowerInt$[3, 6]$ |
| gru_units | PowerInt$[3, 6]$ |
| rec_dims | PowerInt$[3, 5]$ |
| gen_layers | Fixed 1 |
| rec_layers | Fixed 1 |
| lr | LogUniform$[10^{-5}, 10^{-3}]$ |
| weight_decay | $\{0, \text{LogUniform}[10^{-6}, 10^{-3}]\}$ |

Table 9: Neural Flows hyper-parameter search space.

| Parameter | Distribution |
|---|---|
| latents | PowerInt$[2, 5]$ |
| time_hidden_dim | PowerInt$[2, 4]$ |
| gru_units | PowerInt$[3, 6]$ |
| rec_dims | PowerInt$[3, 6]$ |
| hidden_dim | PowerInt$[3, 6]$ |
| hidden_layers | Int$[1, 3]$ |
| flow_layers | Int$[1, 3]$ |
| lr | LogUniform$[10^{-5}, 10^{-3}]$ |
| weight_decay | $\{0, \text{LogUniform}[10^{-6}, 10^{-3}]\}$ |

Table 10: mTAN hyper-parameter search space.

| Parameter | Distribution |
|---|---|
| num_ref_points | PowerInt$[5, 7]$ |
| latent_dim | PowerInt$[3, 5]$ |
| rec_hidden | PowerInt$[3, 5]$ |
| gen_hidden | PowerInt$[3, 6]$ |
| embed_time | PowerInt$[3, 7]$ |
| enc_num_heads | Fixed 1 |
| dec_num_heads | Fixed 1 |
| lr | LogUniform$[10^{-5}, 10^{-3}]$ |
| weight_decay | $\{0, \text{LogUniform}[10^{-6}, 10^{-3}]\}$ |

Table 11: ModernNCA hyper-parameter search space.

| Parameter | Distribution |
|---|---|
| year_order | $\{0, \text{PowerInt}[1, 7]\}$ |
| month_order | $\{0, \text{PowerInt}[1, 7]\}$ |
| week_order | $\{0, \text{PowerInt}[1, 7]\}$ |
| day_order | $\{0, \text{PowerInt}[1, 7]\}$ |
| trend | {True, False} |
| d_embedding | PowerInt$[3, 10]$ |
| d_block | PowerInt$[6, 11]$ |
| n_blocks | Int$[0, 2]$ |
| dim | PowerInt$[6, 11]$ |
| sample_rate | Uniform$[0.01, 0.5]$ |
| drop_out | Uniform$[0.0, 0.5]$ |
| lr | LogUniform$[10^{-5}, 10^{-1}]$ |
| weight_decay | $\{0, \text{LogUniform}[10^{-6}, 10^{-3}]\}$ |

# B Additional Results and Protocol Details

## B.1 Protocol Details

Table 12 spells out the admissible evidence used in the main protocol transformations. The table makes explicit which observations are available at query time and which sources of information are excluded to preserve a valid prediction protocol. In particular, the recovered WE and EO settings differ from the direct TabReD conversion by restoring station-specific histories or pre-offer transaction sequences before applying time-series models.

Table 12: Admissible evidence in the main protocol transformations.

| **Experiment** | **Query** | **Admissible evidence** | **Excluded evidence** |
|---|---|---|---|
| TabReD direct TS | table row | previous rows in global time order | future labels |
| WE recovered | station-time | same-station past rows | other stations, future rows |
| EO recovered | coupon offer | pre-offer transactions | post-offer transactions |
| ITS extrap. | entity-time | observed past segment | future observations |
| ITS interp. | masked point | unmasked observations | masked target value |

## B.2 PFN-Style Baselines

Tables 13 and 14 add PFN-style checks to the main results. These comparisons complement the ModernNCA instantiation with pretrained tabular predictors. TabPFN v2 achieves the lowest RMSE among the reported methods on SH, but trails the ModernNCA variants on the other seven TabReD tasks. TabPFN-TS (Hoo et al., 2025) has higher RMSE than ModernNCA (+ec) on all four irregular regression settings reported here.

Table 13: Additional PFN-style comparisons. TabPFN v2 is evaluated on TabReD, and TabPFN-TS is evaluated on irregular time-series regression tasks. These results are included to clarify that our main ModernNCA instantiation is not positioned as a universal best model.

| **Method** | HI↑ | EO↑ | HD↑ | SH↓ | CT↓ | DE↓ | MR↓ | WE↓ |
|---|---|---|---|---|---|---|---|---|
| TabPFN v2 | 0.8215 | 0.5660 | 0.5000 | 0.2276 | 0.4858 | 0.5529 | 0.1673 | 1.7076 |
| ModernNCA | 0.9571 | 0.5712 | 0.8487 | 0.2526 | 0.4817 | 0.5523 | 0.1631 | 1.4977 |
| ModernNCA (+c) | 0.9575 | 0.6076 | 0.8488 | 0.2368 | 0.4810 | 0.5523 | 0.1628 | 1.4400 |

Table 14: TabPFN-TS on irregular time-series regression. P-I/U-I denote PhysioNet/USHCN interpolation, and P-E/U-E denote extrapolation.

| **Method** | P-I↓ | U-I↓ | P-E↓ | U-E↓ |
|---|---|---|---|---|
| TabPFN-TS | 0.1079 | 0.1091 | 0.1211 | 0.1163 |
| ModernNCA (+ec) | 0.0987 | 0.0948 | 0.1018 | 0.1072 |

## B.3 Training Time

Table 15 reports training time relative to ModernNCA under the same experimental pipeline. The comparison complements the accuracy tables: the context-based tabular instantiation is not only competitive on selected irregular tasks, but also avoids the substantially higher training cost of several continuous-time and recurrent baselines.

Table 15: Average training time relative to ModernNCA.

| **Method** | ModernNCA | Latent ODE | ACSSM | CRU | mTAN | Neural Flows |
|---|---|---|---|---|---|---|
| Relative time | 100.00% | 262.34% | 144.43% | 713.38% | 365.96% | 503.63% |