TIMENET: AN EXTENSIBLE UNIFIED DATA INFRASTRUCTURE FOR NEXT-GENERATION TEMPORAL FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Temporal Foundation Models (TFMs) aim to generalize across domains, datasets, and tasks. Yet, their development remains constrained by fragmented, task-specific data formats, annotations, and processing pipelines. We introduce TimeNet, an open-source data standard and scalable infrastructure that decouples temporal data from task definitions and represents signals, metadata, annotations, and supervision in a shared, extensible data model. TimeNet supports multimodal signals with regular, irregular, or ordinal time axes and expresses different task families (including classification, forecasting, temporal localization, question answering, generation, and editing) as reusable views over the same recordings. This shared representation enables heterogeneous time-series datasets to be combined for large-scale model training across domains, modalities, and tasks. We demonstrate TimeNet by transcoding datasets with 1.5M task instances spanning diverse domains, modalities, temporal scales, and forms of supervision, while retaining practical I/O performance relative to native formats. TimeNet enables an existing TFM training pipeline to support joint training on a configurable number of heterogeneous datasets through configuration changes alone. We show this capability by training Time Series Language Models (TSLMs) across multiple datasets and tasks, obtaining a 14% F1 score improvement compared with models trained on individual datasets. These results show that TimeNet provides the data and systems foundation needed to move beyond task- and dataset-specific TFMs toward models that can learn jointly across heterogeneous domains, modalities, temporal scales, and forms of supervision from a common data model.

1 INTRODUCTION

Recent advances in Temporal Foundation Models (TFMs), including Time Series Foundation Models (TSFMs) and Time Series Language Models (TSLMs), have demonstrated that scaling data, models, and compute can enable efficient transfer and zero/few-shot generalization across heterogeneous time-series datasets, tasks, and application domains (Narayanswamy et al., 2026; Yao et al., 2025; Ansari et al., 2025; Goswami et al., 2024; Shi et al., 2024). However, most existing TSFMs remain specialized to individual tasks, such as forecasting, classification, or anomaly detection, or to particular domains (e.g., weather or health). Real-world temporal systems are often multimodal, asynchronous, event-driven, multi-scale, and subject to distribution shifts (Aksu et al., 2024; Garza et al., 2026). Modeling these systems may require the ability to infer the current state of a system, explain salient events (Rastogi et al., 2026), anticipate future trajectories, and reason about how those trajectories may change under different interventions. For example, monitoring a patient in an intensive care unit can involve detecting the onset of an arrhythmia from Electrocardiogram (ECG), explaining the waveform changes that support it, forecasting the patient's hemodynamic trajectory, and reasoning about how that trajectory would change under an intervention. Next-generation TFMs should therefore move beyond task-specific variants toward shared architectures that can model evolving systems across tasks, modalities, and temporal scales.

Building such models presents two core challenges: (1) learning from heterogeneous forms of supervision across pretraining, supervised fine-tuning, instruction tuning, and post-training with verifiable rewards; and (2) integrating datasets that vary in domain, acquisition protocol, temporal

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.

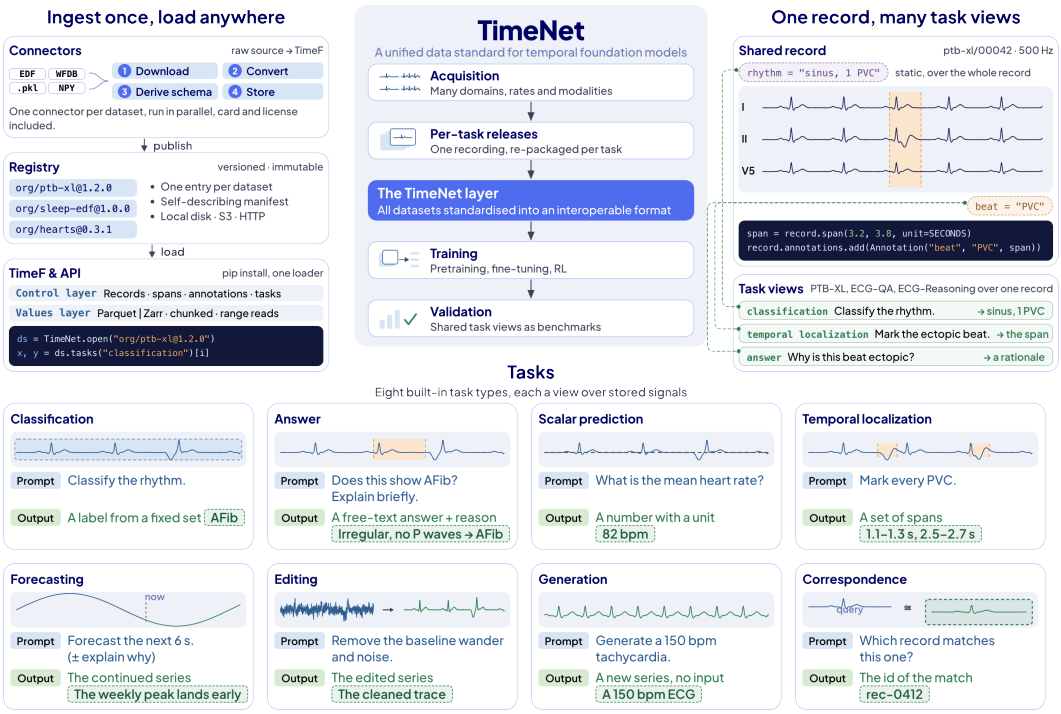

Figure 1: **An Overview of TimeNet.** We introduce a unified data standard, open software infrastructure and an extensive framework for training and evaluation. Multiple tasks can be defined over the same underlying data and across datasets, which can be combined seamlessly during training.

scale, sensor configuration, modality, and tasks. Existing datasets fail to support this vision, since they are typically published as self-contained, task-specific packages with separate and ad-hoc preprocessing pipelines, representations, annotations, and loaders. When new tasks are introduced, the same underlying recordings are often repackaged into new task-specific datasets with custom formats, rather than extending a shared data model. PTB-XL (Wagner et al., 2020), for example, was originally released for ECG classification and later repurposed for question answering in ECG-question answering (QA) (Oh et al., 2023), clinical reasoning in ECG-Reasoning-Benchmark (Oh et al., 2026), and instruction tuning in PULSE (Liu et al., 2024b), each effort introducing its own task formulation and representation of the same underlying ECG recordings. These efforts add supervision to existing recordings, but remain specialized rather than building on a shared representation.

To address these fragmentation challenges and provide a shared foundation for the research field, we introduce **TimeNet**, a unified and extensible data standard and infrastructure for next-generation TFMs. TimeNet makes three core contributions (Figure 1): **(1) A Unified Data Standard:** a common, model-independent format that separates temporal recordings from task definitions, enabling classification, forecasting, anomaly detection, reasoning, generation, and editing as tasks over a shared data representation; **(2) Open Software Infrastructure:** tools for submitting, cataloging, processing, versioning, splitting, and loading datasets through a unified interface; and **(3) An Extensible Foundation for Scalable Training and Benchmarking Across Datasets:** a plug-and-play framework that converts any supported dataset into training tensors without dataset-specific code, enabling to integrate and scale diverse datasets and tasks from a unified representation for large-scale training of multitask TFMs and standardized evaluation. We demonstrate this capability through pooled-dataset training of TSLMs, providing a concrete use case of TimeNet's model-independent infrastructure for TFMs. We open-source all implementations and experiments[1].

[1]Anonymized during review; code attached as supplementary material.

2 Related Work

Time Series resources range from task-specific benchmarks, such as UCR/UEA for classification, Monash for forecasting, and TSB-UAD for anomaly detection (Dau et al., 2019; Bagnall et al., 2018; Godahewa et al., 2021; Paparrizos et al., 2022), to large heterogeneous corpora including Time Series Pile, LOTSA, and Time-MMD (Goswami et al., 2024; Woo et al., 2024; Liu et al., 2024a). Forecasting evaluation has evolved from the M-competitions (Makridakis et al., 2020) toward broader suites such as GIFT-Eval, TSFM-Bench, TIME, BOOM, `fev`, and live evaluation streams such as Impermanent (Aksu et al., 2024; Li et al., 2025; Qiao et al., 2026; Cohen et al., 2025; Shchur et al., 2025; Garza et al., 2026). Recent benchmarks also target reasoning, retrieval, generation, and editing (Cai et al., 2024; Li et al., 2026; Zumarraga et al., 2026a;b; Langer et al., 2025; Shuai et al., 2026; Qiu et al., 2025; Kim et al., 2026; Kong et al., 2025). Complementary efforts standardize software interfaces, dataset formats, or observational data (Löning et al., 2019; Alexandrov et al., 2020; Herzen et al., 2022; Prabowo et al., 2024; Wu et al., 2026; Lansing et al., 2022; Arnrich et al., 2024), while corpora such as UTSD and LOTSA harmonize numerical data for pretraining (Liu et al., 2024c; Woo et al., 2024). Agentic frameworks such as TimeCopilot (Garza & Rosillo, 2025) provide a unified API across TSFMs, but still rely on upstream datasets in compatible formats.

While these efforts focus on curating and evaluating datasets, few address the infrastructure for storing and serving time series at scale for model training, and many existing solutions target only longitudinal (physiological) waveforms. AtriumDB (Goodwin et al., 2020) collects waveforms from diverse devices and vendors into a unified collection, stores them losslessly using a compact delta-encoding compression scheme, indexes them with metadata for quality control and cohort identification without accessing raw data, and enables rapid programmatic retrieval optimized for parallel and distributed computing. Its open-source SDK provides efficient data loaders with batching strategies tailored to physiological waveforms, and the platform has enabled use cases such as real-time bedside models on high-frequency blood-pressure signals (Nagaraj et al., 2024). However, such infrastructure is limited to a single data type and clinical environment, lacks tasks and first-class text supervision, and cannot publish heterogeneous datasets as versioned, shareable entries.

3 Methods

TimeNet addresses limited interoperability, extensibility, and scalability by providing a unified schema and infrastructure that can represent heterogeneous time-series datasets without losing information. Its design is guided by four key principles: **(1) Unified interface:** datasets are accessed through a common schema and API, enabling a single processing and loading pipeline. **(2) Extensibility:**

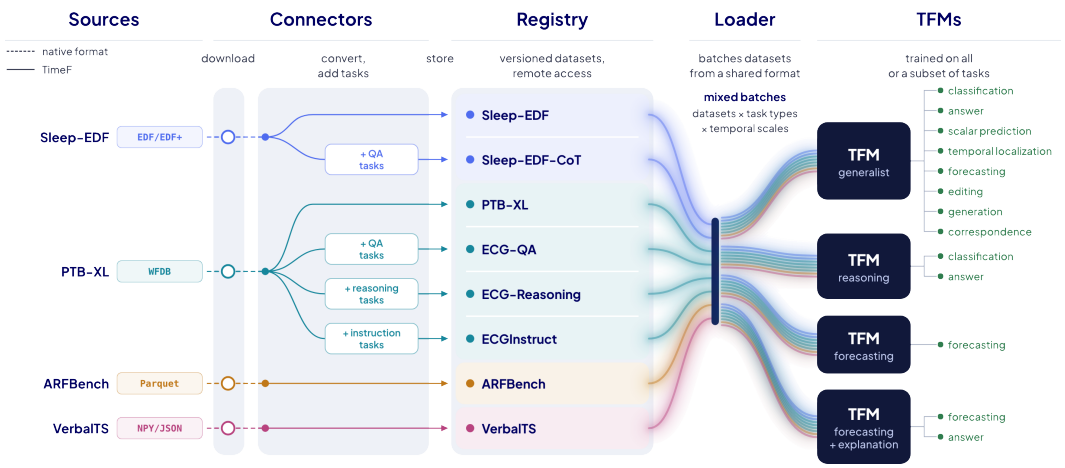

Figure 2: **TimeNet processing pipeline.** TimeNet allows for flexible reuse and recombination of heterogeneous data. Once ingested into TimeNet using a specialized connector, a dataset can be used for any model that operates over applicable Tasks.

records can be augmented with new tasks and annotations without duplicating source data; new task types can be added as needed. **(3) Many-to-many relationships:** tasks and annotations can reference multiple records, and vice versa. **(4) Composability:** records, tasks, and annotations can be independently combined and reused across datasets.

We translate these principles into the various components that form a pipeline, as shown in Figure 2. First, dataset-specific connectors standardize heterogeneous source datasets into the `TimeF` data model, which provides a common representation for signals, annotations, tasks, and their relationships, and stores converted datasets as reusable, versioned entries (Section 3.1). Second, we define a common framework for temporal tasks that makes the structure and relationships between task types explicit. This framework guides TimeNet's built-in task definitions and provides an API through which existing tasks can be specialized or new task types can be added without changing the underlying data representation (Section 3.2). Finally, the data-loading layer operates over the resulting common representation and can select, combine, and collate data from arbitrary datasets into model-specific inputs, independent of their original source format (Section 3.3).

Because dataset ingestion, task definition, and data loading all operate within the same infrastructure and based on a common data model, improvements made at any layer compound across the ecosystem rather than remaining dataset-specific. More generally, TimeNet changes dataset integration with consumers (for example, model architectures or data analysis libraries) from a pairwise problem into an interface problem. With N datasets and M downstream consumers, directly supporting every dataset–consumer combination may require up to $N \times M$ dataset-specific integrations. TimeNet instead requires N dataset connectors and M integrations against the shared TimeF interface, reducing the integration surface to $M + N$. New loading logic, tasks, annotations, batch collation or efficient data distribution can be reused across all compatible datasets, while newly added datasets immediately benefit from the existing infrastructure. As TimeNet grows, both data and functionality therefore accumulate around the same interface, reducing per-dataset integration effort and enabling increasingly large and heterogeneous TFM training.

3.1 Data Representation

We introduce `TimeF` as underlying data model for TimeNet. `TimeF` provides an explicit object hierarchy embedded in a relational representation spanning signals, annotations, and tasks and maps them to an efficient on-disk format. Figure 3 illustrates how the hierarchy maps to the control and value planes of the on-disk format.

Records, Sources, and Signals: In TimeNet, a TimeF-Dataset contains Records, each of which represents a logical observation such as a patient visit, an experimental session, or a machine run. Rather than containing Signals directly, each Record owns one or more Sources that describe where its data originated. Sources form a recursive hierarchy: a Source may contain child Sources as well as Signals, thereby preserving acquisition structures such as devices, or sensor assemblies without encoding provenance in Signal names. Each Signal specifies its data type, shape, unit, and an *axis* along which it is sampled. Axes can be regular (uniform fixed frequency), irregular (non-uniformly sampled), or ordinal (ordered values without timestamps). Signal specifications and axes are immutable and may be shared, while each Signal remains a leaf of a single Record's Source hierarchy. The resulting hierarchy provides a flexible representation of heterogeneous recordings in which the atomic components of a dataset can be composed in arbitrary configurations.

Spans: A Span identifies a point or half-open interval, either in microseconds on a Record timeline or by ordinal position within a Signal (see Figure 3 in green). Time-valued Spans may additionally identify the Signals to which they apply and are used to localize annotations, Task inputs, and temporal targets. This common addressing scheme makes temporal regions reusable across Annotations and Tasks, independently of how the underlying Signals are sampled.

Annotations: Annotations represent typed observations about Datasets, Tasks, Records, Sources, or Signals and may optionally be localized by a Span. TimeF separates reusable annotation content from its occurrences, allowing the same content to be attached to several objects while retaining occurrence-specific provenance, confidence, and metadata. This design allows supervision and metadata to be stored in shared data objects rather than being copied into task-specific dataset variants.

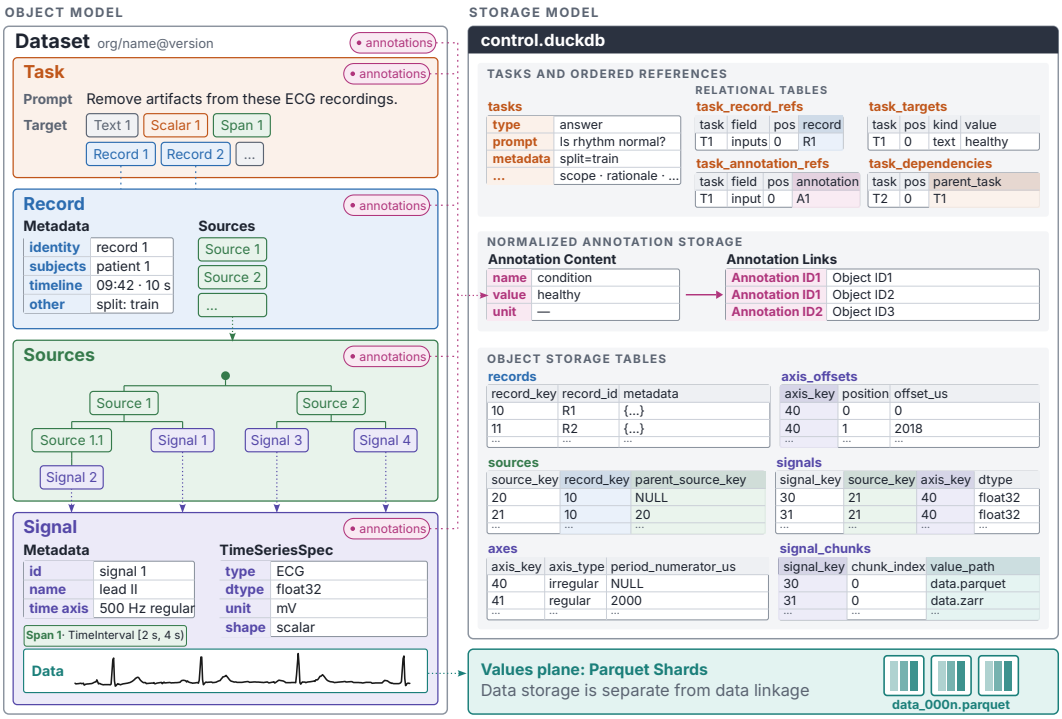

Figure 3: **TimeNet data model.** TimeNet represents each dataset as Records with recursive hierarchies of Sources and Signals. A Source describes where data originated and can contain child Sources. Signals describe value streams and link them to stored values and metadata. Tasks reference one or more Records, and each Record can participate in multiple Tasks. Datasets, Tasks, Records, Sources, and Signals can all have annotations. *DuckDB* stores the hierarchy, annotations, and many-to-many relationships. Underlying Signal values are stored separately in efficient, sharded binary files.

Tasks: TimeNet defines eight task types: *classification*, *question answering*, *scalar prediction*, *temporal localization*, *forecasting*, *generation*, *editing*, and *correspondence*. Tasks reference ordered tuples of input Records and may encode their targets either inline or through references to annotation occurrences, thereby avoiding duplication of the underlying data and supervision. Consequently, the same recordings can support multiple forms of supervision and can be recombined across training and evaluation settings without repackaging the underlying Signals (see Figure 2). Further details and API examples are provided in Appendix A.1.

Object and storage models: TimeNet separates its user-facing object model from the concrete storage model used by TimeF. The object model described above provides an abstract representation of a dataset through Records, Sources, Signals, Annotations, Tasks, and the relationships among them, and can be used through an application programming interface (API). TimeF maps these objects and relationships to an immutable storage representation comprising a JSON *manifest*, a *control plane* database, and the *values plane*; a collection of Signal value files. Within this representation, the control plane is managed with DuckDB, a portable analytical SQL database engine. The database contains relational tables storing the object hierarchy, axes, annotations, Tasks, and their references. The larger Signal arrays containing actual data form the values plane and are written as *Parquet* format. A chunk index in the control plane links each Signal to its physical values, thereby connecting the abstract object model to the underlying data without embedding those arrays in the relational database. This mapping permits relational filtering over metadata and object types without scanning large arrays, while retaining efficient parallel and partial access to Signal values.

Dataset Ingestion and Registry We provide dataset-specific connectors that transform source datasets into TimeNet via a `download`→`convert`→`derive_schema`→`store` pipeline, enabling parallel record processing before dataset-level aggregation. Converted datasets can be published as

immutable, versioned entries in the **TimeNet registry**, identified by `org/name[@version]`, subject to dataset licensing. Restricted, non-redistributable datasets can instead be rebuilt locally using a publishable connector. The registry currently contains 1.51M task instances: 899k question answering, 484k classification, 131k generation, 441 temporal localization, and 292 scalar prediction.

3.2 Formal Specification of Temporal Tasks

Time-series tasks are commonly organized by names such as *classification*, *forecasting*, *anomaly detection*, or *question answering*. However, these names often mix several distinct aspects of a prediction task: *what* the model must predict, *where* that target lies relative to the observed signal, and *how* the task is presented to the model. For example, an anomaly may be represented either as a record-level label or as a temporal interval, while the same prediction may be requested through a structured output or a natural-language question. As new formulations emerge, the same underlying recordings are therefore frequently repackaged into separate task-specific datasets (Section 1). In TimeNet, we instead derive tasks compositionally from their inputs, targets, and temporal relationships.

Task representation. A record r contains Signals $x_k : \mathcal{T}_k \to \mathbb{R}^{c_k}$, $k \in K_r$, together with annotations $\mathcal{A}_r$. A task selects an input view W_{in} over one or more records, optional conditioning C, and a target of type O associated with support W_{out}. The target may be stored directly or derived from existing signals and annotations. The complete TimeNet task specification, including interface, rationale, access protocol, and scoring, is given in Appendix B.

To characterize the semantic operation performed by a task, we retain three properties of this specification. Let $\kappa(O) \in \{\textsc{Value}, \textsc{Span}, \textsc{Signal}, \textsc{Index}\}$ denote the kind of target. Let

$$\Delta(\tau) \in \{\textsc{Obs}, \textsc{Gap}, \textsc{Fut}, \textsc{Past}, \textsc{Chan}, \textsc{Null}\}$$

describe where the target lies relative to the observed input: on observed support, within a missing region, beyond the observation in the future or past, in an unobserved channel, or with no observed input, respectively. Finally, $\alpha(\tau) \in \{1, \geq 2\}$ denotes the number of temporal objects over which the target is jointly defined. We call

$$\phi(\tau) = \big(\kappa(O), \Delta(\tau), \alpha(\tau)\big) \tag{1}$$

the *task signature*.

Different temporal tasks can be constructed by combining the three components of the signature: output kind κ, target position Δ, and arity α. For example, a single-object Value target within the recorded support gives `attribute inference`, whereas a Span target gives `temporal localization`. Moving the target beyond the observed context gives `future-directed inference`, while defining it jointly over multiple temporal objects gives `relational inference`. Signal-valued targets correspond to `temporal construction` or `temporal transformation` depending on whether an observed temporal input is available. These combinations give the six constructions in Table 1, grouped into five semantic task families. The complete signature grid and formal definitions are given in Appendix B.3–B.4. This factorization resolves ambiguity in task names. For instance, a record-level anomaly decision, $X \to \{0, 1\}$, has the signature $(\textsc{Value}, \textsc{Obs}, 1)$ and is therefore attribute inference, whereas identifying anomalous intervals, $X \to \{[t_s, t_e]\}$, has the signature $(\textsc{Span}, \textsc{Obs}, 1)$ and is temporal localization. Likewise, regression and forecasting may both produce numerical targets, but only the latter places the target beyond the observed context.

Task specialization. A task signature specifies the broad semantic family, while more specific task types can be created by imposing additional constraints. For example, ECG rhythm classification specializes classification by restricting the input modality and label space. We denote the set of such task-specific restrictions by $P(\tau)$, which may constrain the admissible inputs, conditioning information, outputs, or temporal support. We write

$$\tau_a \preceq \tau_b \quad \Longleftrightarrow \quad \phi(\tau_a) = \phi(\tau_b) \ \wedge \ P(\tau_b) \subseteq P(\tau_a), \tag{2}$$

so that ECG rhythm classification $\preceq$ classification and anomaly localization $\preceq$ temporal localization. TimeNet records such relationships between tasks, allowing existing task types to be specialized

Table 1: Compositional taxonomy of temporal tasks. Each family is a region of the signature space $\phi = (\kappa, \Delta, \alpha)$ (Definitions 1–2); the six regions partition all well-posed signatures (Appendix B.3). $*$: any value; $\Delta_{\text{in}} = \{\text{OBS}, \text{GAP}, \text{CHAN}\}$, i.e., the target lies within the recorded extent, whether observed, masked, or in an unmeasured channel. TimeNet realizations in `monospace`; +SHIFT etc. denote constructors (Definition 3). For future-directed inference, "other types" are `classification`, `scalar prediction`, and `temporal localization`.

Task family & construction	Signature ϕ	Defining relation	Derived tasks	TimeNet realization
Attribute inference $f(X, C) \to y$	$(\text{VALUE}, \Delta_{\text{in}}, 1)$	Infer a property of the recorded object or a span of it.	Classification, regression, state and severity estimation, captioning	`classification`, `scalar prediction`, `answer`
Temporal localization $f(X, q) \to \{s_i\}$, $s_i \subseteq \mathcal{T}$	$(\text{SPAN}, \Delta_{\text{in}}, 1)$	Identify where or when a property or event occurs in the recorded extent.	Event and anomaly localization, change-point detection, segmentation	`temporal localization`
Future-directed inference $f(X_{\leq t}, C) \to Y_{>t}$	$(*, \{\text{FUT}, \text{PAST}\}, 1)$	Infer a target whose support lies beyond the observed context.	Forecasting, event and future-state prediction, time-to-event, onset prediction	`forecasting`; other types +SHIFT
Relational inference $f(X_i, X_j, C) \to r$	$(*, *, \geq 2)$	Infer a relation between two or more temporal objects or spans.	Ordering, comparison, alignment, similarity, retrieval	`correspondence`; any type +PAIR
Temporal construction $f(C) \to X'$	$(\text{SIGNAL}, \text{NULL}, 1)$	Construct a new temporal object from conditioning alone.	Conditional and unconditional generation	`generation`
Temporal transformation $f(X, C) \to X'$	$(\text{SIGNAL}, \Delta_{\text{in}}, 1)$	Produce a temporal object by transforming or completing a recorded one.	Editing, denoising, reconstruction, imputation, virtual sensing	`editing`; `editing` +MASK

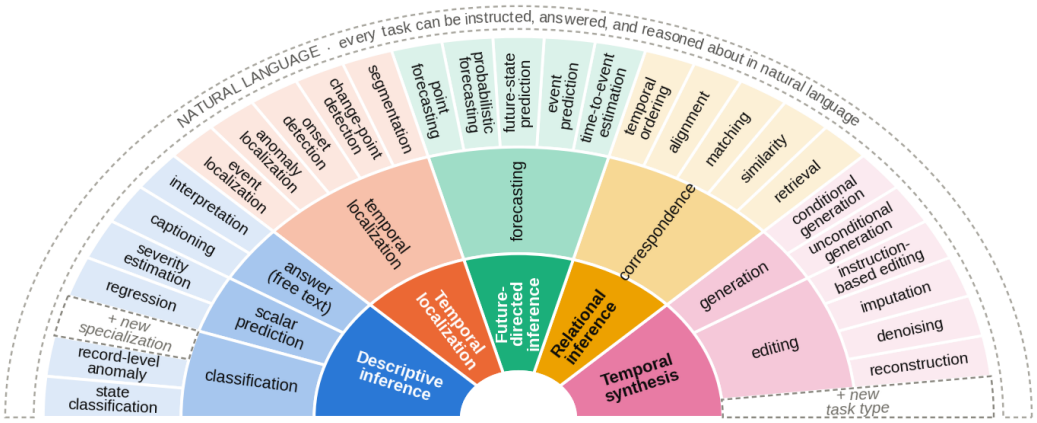

Figure 4: **TimeNet task taxonomy.** TimeNet has five semantic task families, each containing built-in task types that can be further specialized or expanded. Tasks may be expressed in natural language.

without changing the underlying data schema. An overview of how task families map to built-in TimeNet task types and their specializations is shown in Figure 4. The same task may be exposed through a structured interface, natural-language question, or instruction; may optionally require a rationale; and may be evaluated offline or in streaming mode. We formalize these axes, together with the task constructors and their validity conditions, in Appendix B.7–B.10.

3.3 DATA COLLATION

Once datasets are represented in TimeF, the same loading and collation pipeline can be applied independent of their original source format. For a batch of Tasks $\mathcal{B} = \{\tau_i\}_{i=1}^{B}$, TimeNet resolves the Records, Signals, Annotations, temporal scopes, and targets referenced by each τ_i, and materializes only the required Signal values. A model-specific collation function $\mathcal{C}_m(\mathcal{B}) = \big(X_m, M_m, C_m, Y_m\big)$ then maps the resolved task instances into the representation expected by model m, where X_m con-

tains the time-series inputs, M_m optionally identifies valid or observed values, C_m contains additional conditioning such as text or metadata, and Y_m contains the training targets. Figure 5 visualizes this process. Depending on the model, this step may include windowing, padding, resampling, channel

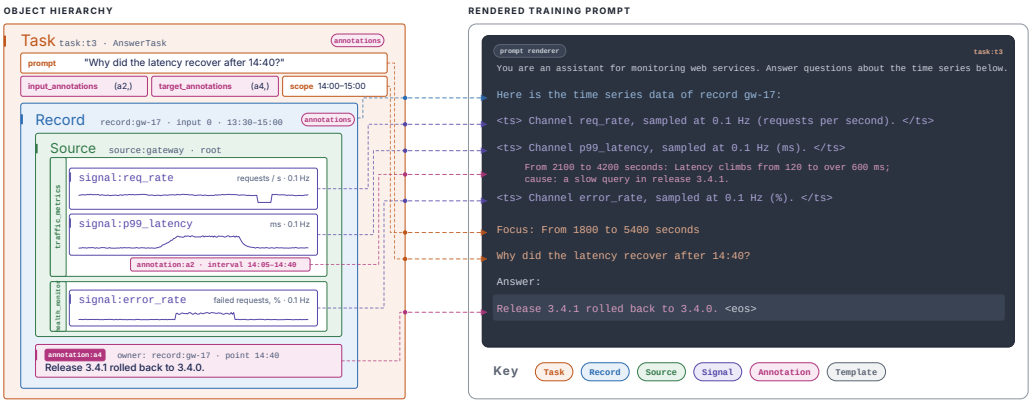

Figure 5: **Data collation in TimeNet.** The structure of TimeNet allows for a common mapping from Tasks to prompts that can be used during training.

selection, or conversion into one or more tensors. Because $\mathcal{C}_m$ operates on the common TimeF objects rather than dataset-specific formats, the same batching and tensorization logic can be reused across datasets and task types, and newly added datasets can directly use the existing training infrastructure, enabling to scale up datasets without having to write extra code.

4 Experiments

To evaluate TimeNet, we investigate whether training jointly on multiple datasets improves performance overall (Section 4.1). We also investigate the input/output speed of accessing heterogeneous datasets through a common representation (Section 4.2).

4.1 Multi-dataset Training

Any TimeNet dataset can be converted into training tensors by the same component, enabling pooled training across datasets. We evaluate whether greater data diversity improves performance using Sleep-EDF Expanded CoT (Kemp et al., 2000; Langer et al., 2025), ECG-QA CoT (Oh et al., 2023; Langer et al., 2025), HEARTS (Li et al., 2026), and ARFBench (Xie et al., 2026), which all support question answering. We conduct this comparison for SLIP Chen et al. (2026) and OpenTSLM Langer et al. (2025), training each model family either separately on each dataset (*SLIP Single*, *OpenTSLM Single*) or jointly on all four (*SLIP Pooled*, *OpenTSLM Pooled*). We use the Llama 3.2-1B Flamingo variant for OpenTSLM and the 400M-parameter model for SLIP. Within each family, single-dataset and pooled models use the same architecture and training configuration. All models are evaluated on the same dataset-specific test sets using F1 score (F1) and accuracy. For OpenTSLM, pooled training

Table 2: OpenTSLM and SLIP performance when trained on each dataset individually versus jointly.

	Sleep-EDF		ECG-QA		HEARTS		ARFBench		Average	
Training	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**
OpenTSLM Single	37.49	77.34	31.19	67.19	15.03	35.71	32.94	**33.33**	29.16	53.39
OpenTSLM Pooled	**41.80**	**80.47**	**34.21**	**73.44**	**19.70**	**38.72**	**37.41**	32.82	**33.28**	**56.36**
SLIP Single	31.78	87.16	**33.93**	**40.01**	9.98	31.07	**20.51**	26.92	24.05	46.29
SLIP Pooled	**42.27**	**87.87**	30.00	31.32	**24.78**	**45.61**	20.38	**29.75**	**29.36**	**48.64**

increases F1 on all four datasets: from 37.49 to 41.80 on Sleep-EDF, 31.19 to 34.21 on ECG-QA,

15.03 to 19.70 on HEARTS, and 32.94 to 37.41 on ARFBench, raising the average F1 from 29.16 to 33.28. Accuracy also increases on Sleep-EDF (77.34 to 80.47), ECG-QA (67.19 to 73.44), and HEARTS (35.71 to 38.72), while decreasing slightly on ARFBench (33.33 to 32.82), resulting in an average accuracy increase from 53.39 to 56.36. For SLIP, performance rises on Sleep-EDF and HEARTS, and accuracy increases on ARFBench, but drops slightly on ECG-QA, from 33.93 to 30.00 F1 and from 40.01 to 31.32 accuracy.

4.2 I/O Performance Across Heterogeneous Datasets

We benchmark TimeNet on six heterogeneous datasets: Sleep-EDF Expanded CoT (Kemp et al., 2000; Langer et al., 2025), ECG-QA CoT (Oh et al., 2023; Langer et al., 2025), PTB-XL (Wagner et al., 2020), HEARTS (Li et al., 2026), ARFBench (Xie et al., 2026), VerbalTS (Gu et al., 2025), and SLIP (Chen et al., 2026). We compare the original data format with the TimeNet version across pandas and PyTorch consumers (see Appendix A.2). Where available, we use original loaders; otherwise, we implement equivalent loaders and verify parity before benchmarking. We provide results in Table 3. TimeNet is not strictly faster than native formats, which often rely on highly-optimized, data-specific, and memory-mapped encodings. However, TimeNet performance remains in comparable ranges, while providing greater flexibility and reusability.

Table 3: Results of I/O performance assessment across 6 datasets.

Dataset	Representation	Consumer	Canonical item count [items]	Time to first item [s]	Full-read time [s]	Sequential throughput [items/s]	Decoded payload throughput [MB/s]	Transferred HTTP payload [GB]	HTTP request count [requests]	Peak resident set size [MiB]
HEARTS	Original	pandas	1,005	**0.275**	93.50	10.7	9.93	1.264	1,008	**397.3**
		PyTorch	1,005	**0.247**	95.30	10.5	9.74	1.264	1,008	**395.5**
	TimeF	*pandas*	*1,005*	*0.574*	***18.43***	***54.5***	***50.38***	***0.331***	***220***	*879.1*
		PyTorch	*1,005*	*0.488*	***17.35***	***57.9***	***53.50***	***0.331***	***220***	*888.2*
ARFBench	Original	pandas	750	**0.479**	64.70	11.6	4.42	0.227	1,010	860.5
		PyTorch	750	**0.366**	59.51	12.6	4.70	0.227	1,010	866.7
	TimeF	*pandas*	*750*	*0.833*	***11.40***	***65.8***	***25.06***	***0.062***	***131***	***808.6***
		PyTorch	*750*	*0.749*	***10.60***	***70.7***	***26.38***	***0.062***	***131***	***827.8***
VerbalTS	Original	pandas	104,628	3.407	**53.74**	**1,946.8**	**11.05**	0.675	146	**958.5**
		PyTorch	104,628	2.934	**15.15**	**6,907.4**	**37.01**	0.674	**140**	**956.2**
	TimeF	*pandas*	*104,628*	***1.087***	*70.68*	*1,480.3*	*8.40*	***0.431***	***144***	*1,036.8*
		PyTorch	*104,628*	***1.129***	*37.04*	*2,824.8*	*15.13*	***0.431***	*144*	*1,011.0*
SLIP	Original	pandas	618,508	**2.841**	**311.54**	**1,985.3**	**21.94**	4.991	**217**	2,226.1
		PyTorch	618,508	**2.415**	**116.47**	**5,310.6**	**57.86**	4.991	**217**	2,457.8
	TimeF	*pandas*	*618,508*	*9.764*	*469.03*	*1,318.7*	*14.57*	***3.984***	*1,656*	***2,036.1***
		PyTorch	*618,508*	*9.830*	*282.78*	*2,187.3*	*23.83*	***3.984***	*1,656*	***2,033.9***
ECG-QA	Original	pandas	231,543	28.787	2,735.10	84.7	20.53	3.267	32,305	2,139.6
		PyTorch	231,543	32.103	2,351.40	98.5	23.80	3.267	32,305	2,139.0
	TimeF	*pandas*	*231,543*	***1.740***	***940.21***	***246.3***	***59.72***	***2.048***	***2,070***	***1,150.5***
		PyTorch	*231,543*	***1.837***	***644.28***	***359.4***	***86.88***	***2.048***	***2,070***	***1,168.7***
Sleep-EDF	Original	pandas	458,161	**0.718**	**381.05**	**1,202.4**	**45.91**	8.714	**197**	**786.8**
		PyTorch	458,161	**0.751**	**170.83**	**2,682.0**	**100.91**	8.714	**197**	**760.0**
	TimeF	*pandas*	*458,161*	*1.767*	*633.64*	*723.1*	*27.61*	***5.616***	*4,204*	*1,081.8*
		PyTorch	*458,161*	*1.778*	*431.64*	*1,061.4*	*39.94*	***5.616***	*4,204*	*1,089.4*

5 Discussion

Principal findings. TimeNet provides a common representation and infrastructure for heterogeneous temporal datasets that differ in domain, modality, sampling structure, and task formulation. Across six datasets comprising more than 1.5 million task instances, the same interface supports data access and model-specific collation without dataset-specific loading pipelines, while maintaining practical I/O performance relative to native formats. This shared representation also enables datasets to be combined directly for training: for OpenTSLM, pooled training increases F1 on all four datasets. SLIP similarly improves on Sleep-EDF and HEARTS, but shows a substantial decrease on ECG-QA. The slight performance drop on ECG-QA might be explained by SLIPs smaller model size (400M parameters) compared to OpenTSLM 1B parameters, giving it less capacity to learn across tasks.
Practical implications. We believe the research field must move beyond fragmented, one-off dataset releases toward publishing datasets into shared registries, with common means to read, write, annotate, expand, and version data, as provided by TimeNet. Recordings should be stored once, while new annotations and tasks can be added over time without duplicating or restructuring the original data. Toward this end, we have begun by combining heterogeneous data into a public registry.
Limitations and future work. While TimeNet's data representation and interfaces are generally model- and task- independent, our training experiments evaluate only selected models and tasks rather than the full range of TFM families. Our current evaluation focuses on data access across six datasets and training across four datasets, and, hence, only evaluates a subset of dataset and task families supported by TimeNet. Additionally, we only assesses training on a limited number of task types: Question answering and time series-based reasoning, but leave training of a model that spans all tasks to future work.

Reproducibility Statement

All source code associated with this work is publicly available. All external datasets used are open source. These resources ensure full reproducibility of our results.

Use of AI

In this work, we used generative AI tools to implement parts of the software code used in our experimental pipeline and to assist with the formal expression of portions of our task definitions, including mathematical notation and formulation. The underlying concepts, task taxonomy, and scientific content of these definitions were developed by the authors. We have not used generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, provide critical ingredients for or assist in writing mathematical proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean or reformat datasets, support qualitative or thematic data analysis, interpret results, or draw scientific conclusions. Additionally, we used generative AI tools to edit portions of the manuscript to improve grammar, clarity, and readability; to review source code for potential errors or bugs; and to create or modify scientific figures or images.

We have reviewed all AI-assisted work. In particular, AI-assisted code was developed under strict human supervision, reviewed by at least two authors through pull requests before being accepted into the repository, and tested for correctness before being used in experiments. AI-assisted text, mathematical formulations, and visual materials were likewise reviewed and approved by the authors.

We take responsibility for the final content of this work, including text, claims, code, figures, and other artifacts produced or modified with the aid of generative AI.

References

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. GIFT-Eval: A benchmark for general time series forecasting model evaluation. *arXiv e-prints*, October 2024. URL https://arxiv.org/abs/2410.10393.

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. GluonTS: Probabilistic and neural time series modeling in Python. *J. Mach. Learn. Res.*, 21(116):1–6, 2020. URL http://jmlr.org/papers/v21/19-820.html.

James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26 (11):832–843, 1983. doi: 10.1145/182.358434.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. *arXiv e-prints*, October 2025. URL https://arxiv.org/abs/2510.15821.

Bert Arnrich, Edward Choi, Jason Fries, Matthew McDermott, Jungwoo Oh, Tom Pollard, Nigam Shah, Ethan Steinberg, Michael Wornow, and Robin van de Water. Medical event data standard (MEDS): Facilitating machine learning for health. In *Int. Conf. Learn. Representations Workshops (ICLRW)*, 2024. URL https://iclr.cc/virtual/2024/23574. Workshop on Learning from Time Series for Health.

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018. *arXiv e-prints*, November 2018. URL https://arxiv.org/abs/1811.00075.

Yifu Cai, Arjun Choudhry, Mononito Goswami, and Artur Dubrawski. TimeSeriesExam: A time series understanding exam. *arXiv e-prints*, October 2024. URL https://arxiv.org/abs/2410.14752.

Yuliang Chen, Arvind Pillai, Yu Yvonne Wu, Tess Z. Griffin, Lisa Marsch, Michael V. Heinz, Nicholas C. Jacobson, and Andrew Campbell. Learning transferable sensor models via language-informed pretraining. *arXiv e-prints*, March 2026. URL https://arxiv.org/abs/2603.11950.

Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean Ogier du Terrail, Anna-Monica Toon, Kan Wang, Stephan Xie, Zongzhe Xu, Viktoriya Zhukova, David Asker, Ameet Talwalkar, and Othmane Abou-Amal. This time is different: An observability perspective on time series foundation models. *arXiv e-prints*, May 2025. URL https://arxiv.org/abs/2505.14766.

Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The UCR time series archive. *IEEE/CAA J. Autom. Sinica*, 6(6):1293–1305, 2019. doi: 10.1109/JAS.2019.1911747.

Azul Garza and Renée Rosillo. TimeCopilot. *arXiv e-prints*, August 2025. URL https://arxiv.org/abs/2509.00616.

Azul Garza, Renée Rosillo, Rodrigo Mendoza-Smith, David Salinas, Andrew Robert Williams, Arjun Ashok, Mononito Goswami, and José Martín Juárez. Impermanent: A live benchmark for temporal generalization in time series forecasting. *arXiv e-prints*, March 2026. URL https://arxiv.org/abs/2603.08707.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive. In *Conf. Neural Inf. Process. Syst. Datasets and Benchmarks Track*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/eddea82ad2755b24c4e168c5fc2ebd40-Abstract-round2.html.

Andrew J Goodwin, Danny Eytan, Robert W Greer, Mjaye Mazwi, Anirudh Thommandram, Sebastian D Goodfellow, Azadeh Assadi, Anusha Jegatheeswaran, and Peter C Laussen. A practical approach to storage and retrieval of high-frequency physiological signals. *Physiological measurement*, 41(3):035008, 2020.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In *Proc. Int. Conf. Mach. Learning (ICML)*, volume 235, pp. 16115–16152, 2024. URL https://proceedings.mlr.press/v235/goswami24a.html.

Shuqi Gu, Chuyue Li, Baoyu Jing, and Kan Ren. VerbalTS: Generating time series from texts. In *Proc. Int. Conf. Mach. Learning (ICML)*, volume 267, pp. 20448–20476, 2025. URL https://proceedings.mlr.press/v267/gu25a.html.

Julien Herzen, Francesco Lässig, Samuele Giuliano Piazzetta, Thomas Neuer, Léo Tafti, Guillaume Raille, Tomas Van Pottelbergh, Marek Pasieka, Andrzej Skrodzki, Nicolas Huguenin, Maxime Dumonal, Jan Kościsz, Dennis Bader, Frédérick Gusset, Mounir Benheddi, Camila Williamson, Michal Kosinski, Matej Petrik, and Gaël Grosch. Darts: User-friendly modern machine learning for time series. *J. Mach. Learn. Res.*, 23(124):1–6, 2022. URL http://jmlr.org/papers/v23/21-1177.html.

Bob Kemp, Aeilko H. Zwinderman, Bert Tuk, Hilbert A. C. Kamphuisen, and Josefien J. L. Oberye. Analysis of a sleep-dependent neuronal feedback loop: The slow-wave microcontinuity of the EEG. *IEEE Trans. Biomed. Eng.*, 47(9):1185–1194, September 2000. doi: 10.1109/10.867928.

Jaeho Kim, Changhun Oh, Seokhyun Lee, Irina Rish, and Changhee Lee. Representing time series as structured programs for llm reasoning. *arXiv e-prints*, June 2026. URL https://arxiv.org/abs/2606.12481.

Yaxuan Kong, Yiyuan Yang, Yoontae Hwang, Wenjie Du, Stefan Zohren, Zhangyang Wang, Ming Jin, and Qingsong Wen. Time-MQA: Time series multi-task question answering with context enhancement. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher

Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29736–29753, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1437. URL https://aclanthology.org/2025.acl-long.1437/.

Patrick Langer, Thomas Kaar, Max Rosenblattl, Maxwell A. Xu, Winnie Chow, Martin Maritsch, Aradhana Verma, Brian Han, Daniel Seung Kim, Henry Chubb, Scott Ceresnak, Aydin Zahedivash, Alexander Tarlochan Singh Sandhu, Fatima Rodriguez, Daniel McDuff, Elgar Fleisch, Oliver Aalami, Filipe Barata, and Paul Schmiedmayer. OpenTSLM: Time-series language models for reasoning over multivariate medical text- and time-series data. *arXiv e-prints*, October 2025. URL https://arxiv.org/abs/2510.02410.

Carina Lansing, Maxwell Levin, Chitra Sivaraman, Rebecca Fao, and Frederick Driscoll. Tsdat: An open-source data standardization framework for marine energy and beyond. In *MTS/IEEE OCEANS Conf.*, San Diego, CA, USA, 2022. doi: 10.23919/OCEANS44145.2021.9706101.

Sirui Li, Shuhan Xiao, Mihir Joshi, Ahmed Metwally, Daniel McDuff, Wei Wang, and Yuzhe Yang. HEARTS: Benchmarking LLM reasoning on health time series. *arXiv e-prints*, March 2026. URL https://arxiv.org/abs/2603.06638.

Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, and Bin Yang. TSFM-Bench: A comprehensive and unified benchmark of foundation models for time series forecasting. In *ACM SIGKDD Int. Conf. Knowl. Discovery and Data Mining (KDD)*, 2025. doi: 10.1145/3711896.3737442. URL https://arxiv.org/abs/2410.11802.

Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Kamarthi, Aditya B. Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, Chao Zhang, and B. Aditya Prakash. Time-MMD: Multi-domain multimodal dataset for time series analysis. In *Conf. Neural Inf. Process. Syst. Datasets and Benchmarks Track*, volume 37, 2024a. doi: 10.52202/079017-2476. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/8e7768122f3eeec6d77cd2b424b72413-Abstract-Datasets_and_Benchmarks_Track.html.

Ruoqi Liu, Yuelin Bai, Xiang Yue, and Ping Zhang. Teach multimodal LLMs to comprehend electrocardiographic images. *arXiv e-prints*, October 2024b. URL https://arxiv.org/abs/2410.19008.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *Proc. Int. Conf. Mach. Learning (ICML)*, volume 235, pp. 32369–32399, 2024c. URL https://proceedings.mlr.press/v235/liu24cb.html.

Markus Löning, Anthony Bagnall, Sajaysurya Ganesh, Viktor Kazakov, Jason Lines, and Franz J. Király. sktime: A unified interface for machine learning with time series. *arXiv e-prints*, September 2019. URL https://arxiv.org/abs/1909.07872.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 competition: 100,000 time series and 61 forecasting methods. *Int. J. Forecasting*, 36(1):54–74, 2020. doi: 10.1016/j.ijforecast.2019.04.014. URL https://www.sciencedirect.com/science/article/pii/S0169207019301128. M4 Competition.

Sujay Nagaraj, Andrew J Goodwin, Dmytro Lopushanskyy, Sebastian David Goodfellow, Danny Eytan, Hadrian Balaci, Robert Greer, Anand Jayarajan, Azadeh Assadi, Mjaye Leslie Mazwi, et al. Needles in needle stacks: Meaningful clinical information buried in noisy sensor data. In *Machine Learning for Healthcare Conference*. PMLR, 2024.

Girish Narayanswamy, Maxwell A. Xu, A. Ali Heydari, Samy Abdel-Ghaffar, Marius Guerard, Kara Vaillancourt, Zhihan Zhang, Jake Garrison, Levi Albuquerque, Dimitris Spathis, Hong Yu, Hamid Palangi, Xuhai Xu, David G. T. Barrett, Joseph Breda, Jed McGiffin, Yubin Kim, Yuwei Zhang, Naghmeh Rezaei, Samuel Solomon, Karan Ahuja, Tim Althoff, Jake Sunshine,

Ming-Zher Poh, Benjamin Yetton, Ari Winbush, Nicholas B. Allen, James M. Rehg, Isaac Galatzer-Levy, Yun Liu, John Hernandez, Anupam Pathak, Conor Heneghan, Yuzhe Yang, Ahmed A. Metwally, Pushmeet Kohli, Mark Malhotra, Shwetak Patel, Xin Liu, and Daniel McDuff. Towards a general intelligence and interface for wearable health data. *arXiv e-prints*, May 2026. URL https://arxiv.org/abs/2605.22759.

Jungwoo Oh, Gyubok Lee, Seongsu Bae, Joon myoung Kwon, and Edward Choi. ECG-QA: A comprehensive question answering dataset combined with electrocardiogram. In *Conf. Neural Inf. Process. Syst. Datasets and Benchmarks Track*, volume 36, pp. 66277–66288, 2023.

Jungwoo Oh, Hyunseung Chung, Junhee Lee, Min-Gyu Kim, Hangyul Yoon, Ki Seong Lee, Youngchae Lee, Muhan Yeo, and Edward Choi. ECG-Reasoning-Benchmark: A benchmark for evaluating clinical reasoning capabilities in ECG interpretation. *arXiv e-prints*, March 2026. URL https://arxiv.org/abs/2603.14326.

John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S. Tsay, Themis Palpanas, and Michael J. Franklin. TSB-UAD: An end-to-end benchmark suite for univariate time-series anomaly detection. *Proc. VLDB Endowment*, 15(8):1697–1711, 2022. URL https://www.vldb.org/pvldb/vol15/p1697-paparrizos.pdf.

Arian Prabowo, Xiachong Lin, Imran Razzak, Hao Xue, Emily W. Yap, Matthew Amos, and Flora D. Salim. Bts: Building timeseries dataset: Empowering large-scale building analytics. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=6cCFK69vJI.

Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's TIME: Towards the next generation of time series forecasting benchmarks. *arXiv e-prints*, February 2026. URL https://arxiv.org/abs/2602.12147.

Jiaxing Qiu, Dongliang Guo, Brynne Sullivan, Teague R. Henry, and Tom Hartvigsen. Instruction-based time series editing. *arXiv e-prints*, August 2025. URL https://arxiv.org/abs/2508.01504.

Sparsh Rastogi, Tanmay Kumar, Baiyu Chen, Jatin Bedi, Zechen Li, and Flora D Salim. Trace-ts: Attribution-grounded and traceable sensor-language reasoning for human activity understanding. *arXiv preprint arXiv:2608.00200*, 2026.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting. *arXiv e-prints*, September 2025. URL https://arxiv.org/abs/2509.26468.

Jingzhe Shi, Qinwei Ma, Huan Ma, and Lei Li. Scaling law for time series forecasting. *Advances in Neural Information Processing Systems*, 37:83314–83344, 2024.

Zitao Shuai, Zongzhe Xu, Yuntian Wu, Sirui Li, Tianhong Li, and Yuzhe Yang. Signal or noise? Understanding generative models for real-world sensor time series. *arXiv e-prints*, July 2026. URL https://arxiv.org/abs/2607.04245.

Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima Lunze, Wojciech Samek, and Tobias Schaeffter. PTB-XL, a large publicly available electrocardiography dataset. *Sci. Data*, 7:154, May 2020. doi: 10.1038/s41597-020-0495-6.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Proc. Int. Conf. Mach. Learning (ICML)*, volume 235, pp. 53140–53164, 2024. URL https://proceedings.mlr.press/v235/woo24a.html.

John Wu, Yongda Fan, Zhenbang Wu, Paul Landes, Eric Schrock, Sayeed Sajjad Razin, Arjun Chatterjee, Naveen Baskaran, Joshua Steier, Andrea Fitzpatrick, Bilal Arif, Rian Atri, Jathurshan Pradeepkumar, Siddhartha Laghuvarapu, Junyi Gao, Adam R. Cross, and Jimeng Sun. PyHealth 2.0: A comprehensive open-source toolkit for accessible and reproducible clinical deep learning. *arXiv e-prints*, January 2026. URL https://arxiv.org/abs/2601.16414.

Stephan Xie, Ben Cohen, Mononito Goswami, Junhong Shen, Emaad Khwaja, Chenghao Liu, David Asker, Othmane Abou-Amal, and Ameet Talwalkar. ARFBench: Benchmarking time series question answering ability for software incident response. *arXiv e-prints*, April 2026. URL https://arxiv.org/abs/2604.21199.

Qingren Yao, Chao-Han Huck Yang, Renhe Jiang, Yuxuan Liang, Ming Jin, and Shirui Pan. Towards neural scaling laws for time series foundation models. In *International Conference on Learning Representations*, volume 2025, pp. 83793–83820, 2025.

Nicolas Zumarraga, Thomas Kaar, Ning Wang, Maxwell A. Xu, Max Rosenblattl, Markus Kreft, Kevin O'Sullivan, Paul Schmiedmayer, Patrick Langer, and Robert Jakob. TS-Haystack: A multi-scale retrieval benchmark for time series language models. In *Int. Conf. Learn. Representations Workshops (ICLRW)*, 2026a. URL https://openreview.net/forum?id=vGMayMuH83. Workshop on Time Series in the Age of Large Models.

Nicolas Zumarraga, Lorenzo Steno, Ning Wang, Max Rosenblattl, Thomas Kaar, Maxwell A. Xu, Kevin O'Sullivan, Markus Kreft, Elgar Fleisch, Paul Schmiedmayer, Patrick Langer, and Robert Jakob. TimeRLM: Recursive language models enable precise anomaly localization in long-context time-series. *arXiv e-prints*, August 2026b. URL https://arxiv.org/abs/2608.03391.

A TimeNet API and Implementation Details

A.1 API Example

The TimeNet API exposes the logical object hierarchy directly and hides the physical layout of the underlying TimeF version. A consumer addresses any dataset by an `org/name[@version]` reference, while the client resolves its registry, reconstructs its typed objects from the relational control plane, and leaves Signal values lazy until they are read. The same code therefore searches, loads, and traverses datasets independently of original file formats and source-specific dependencies.

The TimeNet API is declarative and object-oriented. Dataset authors are encouraged to use the object hierarchy directly, rather than providing a separate serialization interface. As discussed in the main paper extensively, a Signal combines values with a typed specification and axis, Sources organize Signals recursively, and Records collect the resulting acquisition hierarchy. Calling `annotate()` attaches reusable annotation content and returns the particular occurrence that a Task can reference as context or supervision. Concrete Task classes then reference ordinary Record and Annotation objects, while `add_record()` and `add_task()` validate ownership, identity, temporal bounds, and cross-object references before modifying the Dataset. Finally, a single `write()` call derives the schema and commits the manifest, DuckDB control plane, and Parquet or Zarr values plane as one immutable version. Users consequently work with a small set of composable Python objects and are not required to interact with the relational layer or data storage directly. Figure 6 illustrates the ease of both consumption and authoring through the current public API.

Signals and spans

```
ds = TimeNet.open(
    "org/dataset@1.0.0")
print(len(ds.records))
record = ds[0]

eeg = record.signal("eeg")
span = eeg.span(
    start=12, end=18,
    unit=UNIT.SECONDS)

x = span.to_numpy()
```

Annotations

```
seizure = Annotation(
    key="seizure",
    value="tonic-clonic",
    provenance="neurologist",
    span=span)

record.annotations.add(seizure)
```

Tasks

```
clsTask = ClassificationTask(
    input=span,
    target=seizure)

qaTask = AnswerTask(
    prompt="What occurs here?",
    input=span,
    target="Tonic-clonic seiz.",
    parents=[clsTask])

record.add_tasks(clsTask, qaTask)
```

Figure 6: TimeNet API: loading signals and defining spans, attaching annotations, and deriving tasks.

A.2 Datasets

Table 4: The datasets, as released. Task types name the tags of TimeF's task registry.

Dataset	Size	Released format	Domains	Task types	Overlap
Sleep-EDF Exp.	8.1 GB	EDF, EDF+ (int16), XLS	sleep, EEG, EOG, EMG, respiration	scalar prediction, temporal localization	SLIP
ECG-QA CoT	1.89 GB	WFDB (int16), CSV	cardiology, ECG	question answering	SLIP
HEARTS	1.32 GB	pickle (pandas DataFrame, float64)	metabolic, respiration, COVID cough	scalar prediction, forecasting, temporal localization	Capture-24
ARFBench	103 MB	Parquet (datetime64, float64, string, int64), CSV	infrastructure, networking, database, security		none
VerbalTS	800 MB	NPY (float64, UTF-32, int64), JSON	synthetic, energy, traffic, motion, weather	series generation	ETTm1, Weather
SLIP	6.14 GB	Parquet (nested float32, string)	health, web, nature, energy, IoT, transport, activity, stress	question answering	Sleep-EDF, PTB-XL, Capture-24

B A COMPOSITIONAL TAXONOMY OF TEMPORAL TASKS

This appendix gives the formal details behind Section 3.2. We first define the full task specification and support relations, then enumerate the signature space and task families. We subsequently describe the output-type registry, task specialization, orthogonal interaction axes, task constructors, and the current scope of the specification.

B.1 FULL TASK SPECIFICATION

A record r owns a recursive hierarchy of Sources whose leaf Signals form the set K_r. Each signal $x_k : \mathcal{T}_k \to \mathbb{R}^{c_k}$ is defined over a regular, irregular, or ordinal temporal axis. Annotations $\mathcal{A}_r$ may be associated with the Record, its Sources, or its Signals.

The set of channel–time cells available in a record is

$$\Omega_r = \bigcup_{k \in K_r} \{k\} \times \mathcal{T}_k. \tag{3}$$

A *view* is any subset $W \subseteq \Omega_r$. Views may span multiple signals or, for tasks involving several temporal objects, multiple records.

Definition 1 (Task specification)**.** A temporal task is represented analytically as

$$\tau = \big\langle W_{\text{in}},\, C,\, (O, W_{\text{out}}, g),\, \iota,\, \varepsilon,\, \pi,\, \mu \big\rangle, \tag{4}$$

where:

- W_{in} contains the input views over one or more records;
- C contains optional conditioning such as metadata, text, a query, or in-context examples;
- O is an output type from the extensible registry $\mathcal{O}$ (Section B.5);
- W_{out} specifies the temporal support to which the target refers;
- g is a registered deterministic function that derives the target from stored signals and annotations, with identity as the default;
- $\iota \in \{\text{structured}, \text{question}, \text{instruction}\}$ specifies the interaction interface;
- $\varepsilon \in \{\text{none}, \text{text}, \text{evidence}\}$ specifies the rationale requirement;
- $\pi \in \{\text{offline}, \text{streaming}\}$ specifies the access protocol; and
- μ is a scoring function, either per instance $\mu(\hat{y}, y)$ or per corpus $\mu(\{\hat{y}_i\}, \{y_i\})$.

For non-signal targets, W_{out} denotes the temporal support *referenced* by the target rather than a physical location of the output itself. For example, a classification label may describe an ECG interval even though the label itself has no temporal extent.

The tuple above is an analytical representation rather than a literal serialization of the TimeF Task object. Stored Tasks provide references to Records, targets or target Annotations, and optional prompt, scope, rationale, and provenance fields. Other components, such as the access protocol or evaluation metric, may be supplied by the benchmark or model-facing layer.

B.2 SUPPORT RELATION AND ARITY

Let K_{out} denote the target channels and let

$$W_{\text{in}}^{*} = W_{\text{in}} \cap \left(\bigcup_{k \in K_{\text{out}}} \{k\} \times \mathcal{T}_k \right) \tag{5}$$

be the observed portion of those channels.

Definition 2 (Support relation and arity)**.** The target position is

$$\Delta(\tau) \in \{\text{OBS}, \text{GAP}, \text{FUT}, \text{CHAN}, \text{NULL}\}, \tag{6}$$

with:

- OBS: the target refers to support exposed in W_{in};
- GAP: the target lies within the temporal extent of the observed target channels but is withheld from the input;
- FUT: the target lies after the observed support of the target channels;
- CHAN: the target channels are not observed although other input channels are available; and
- NULL: $W_{\text{in}} = \emptyset$.

Independently,

$$\alpha(\tau) \in \{1, \geq 2\} \tag{7}$$

denotes the number of temporal objects over which the target is jointly defined.

Known future covariates do not violate $\Delta = \text{FUT}$: the relation is computed with respect to the target channels rather than all available channels.

Because the exposed input is represented explicitly by W_{in}, targets outside the observed support can be withheld mechanically. Under $\pi =$ streaming, the loader additionally exposes only the prefix available at each prediction time.

B.3 THE SIGNATURE GRID

Let

$$\kappa(O) \in \{\text{VALUE}, \text{SPAN}, \text{SIGNAL}, \text{INDEX}\} \tag{8}$$

denote the semantic kind of the output type. VALUE includes labels, real-valued outputs, and free text. Composite outputs take the most temporally structured kind of their components, using the precedence

$$\text{SIGNAL} \succ \text{SPAN} \succ \text{INDEX} \succ \text{VALUE}. \tag{9}$$

The task signature is

$$\phi(\tau) = \big(\kappa(O), \Delta(\tau), \alpha(\tau)\big). \tag{10}$$

Let

$$\Delta_{\text{in}} = \{\text{OBS}, \text{GAP}, \text{CHAN}\}. \tag{11}$$

Table 5 enumerates the resulting signature space.

Single-object VALUE or SPAN targets with $\Delta = \text{NULL}$ are undefined because there is no temporal object to describe or localize. Likewise, an INDEX output requires a candidate set and is therefore relational.

Proposition 1 (Partition of the signature space)**.** *Every well-posed signature belongs to exactly one of the six constructions in Table 1.*

Table 5: Signature grid for temporal tasks. Rows denote output kind κ and columns denote support relation Δ for single-object tasks. The final column contains relational tasks with $\alpha \geq 2$. [A], [L], [F], [R], [C], and [T] denote attribute inference, temporal localization, future-directed inference, relational inference, temporal construction, and temporal transformation, respectively.

	single object ($\alpha = 1$)					relational
	OBS	GAP	FUT	CHAN	NULL	$\alpha \geq 2$
VALUE	[A] classification, scalar, answer	[A] value over masked region	[F] event prediction	[A] cross-channel inference	—	[R] comparison, ordering
SPAN	[L] localization, segmentation	[L] localization in masked region	[F] onset prediction	[L] cross-channel localization	—	[R] alignment
SIGNAL	[T] editing, denoising	[T] imputation	[F] forecasting	[T] virtual sensing	[C] generation	[R] signal relation
INDEX	—	—	—	—	—	[R] retrieval

Proof. If $\alpha \geq 2$, the task is relational inference. Otherwise $\alpha = 1$. If $\Delta = \text{FUT}$, the task is future-directed inference regardless of output kind. For the remaining support relations, the output kind distinguishes VALUE, SPAN, and SIGNAL targets. Signal targets with $\Delta = \text{NULL}$ are temporal construction; signal targets with $\Delta \in \Delta_{\text{in}}$ are temporal transformation. The remaining valid cases are attribute inference and temporal localization. Ill-posed cells are excluded by definition. □

The two signal-producing constructions—construction and transformation—form the broader *temporal synthesis* family used in Fig. 4. Thus the six constructions induce five semantic families.

B.4 FORMAL DEFINITIONS OF THE CONSTRUCTIONS

Let X denote the selected input views and C their conditioning.

Attribute inference. For

$$\phi(\tau) = (\text{VALUE}, \Delta_{\text{in}}, 1), \tag{12}$$

the task asks for a value describing the recorded object or a supported region,

$$f(X, C) \to y. \tag{13}$$

The codomain distinguishes common task types: $y \in \{c_1, \ldots, c_K\}$ gives classification, $y \in \mathbb{R}^d$ gives scalar prediction, and free text gives `answer`, including captioning or open-ended interpretation.

Temporal localization. For

$$\phi(\tau) = (\text{SPAN}, \Delta_{\text{in}}, 1), \tag{14}$$

the target identifies where or when a phenomenon occurs,

$$f(X, q) \to \{s_1, \ldots, s_n\}, \qquad s_i \subseteq \mathcal{T}, \tag{15}$$

where q may specify the phenomenon. Examples include event localization, anomaly localization, change-point detection, and segmentation.

Future-directed inference. For

$$\phi(\tau) = (*, \text{FUT}, 1), \tag{16}$$

the target lies beyond the observed support of the target channels,

$$f(X_{\leq t}, C) \to Y_{>t}. \tag{17}$$

Forecasting is the signal-valued case. Value-valued targets include future-state and event prediction, while span-valued targets include onset prediction and related time-to-event formulations.

Relational inference. For

$$\phi(\tau) = (*, *, \geq 2), \tag{18}$$

the target is jointly defined over multiple temporal objects,

$$f(X_i, X_j, C) \to r. \tag{19}$$

Examples include comparison, ordering, alignment, similarity, correspondence, and retrieval. Temporal ordering between intervals may use Allen's interval relations (Allen, 1983).

Temporal construction. For

$$\phi(\tau) = (\textsc{Signal}, \textsc{Null}, 1), \tag{20}$$

the task constructs a temporal signal without an observed temporal input,

$$f(C) \to X'. \tag{21}$$

This includes conditional and unconditional generation.

Temporal transformation. For

$$\phi(\tau) = (\textsc{Signal}, \Delta_{\text{in}}, 1), \tag{22}$$

the task produces a temporal signal from an existing recorded object,

$$f(X, C) \to X'. \tag{23}$$

Editing and denoising correspond to observed support, imputation to a gap, and virtual sensing to an unobserved target channel. Together with temporal construction, these tasks form temporal synthesis.

B.5 Output-type registry

TimeNet separates the semantic kind κ from the concrete output type O. Table 6 gives representative built-in output types and default evaluation functions.

A new output type is registered together with a validator, semantic kind, and compatible scoring functions. Composite outputs inherit the highest-precedence kind of their components. For example, labeled temporal segments are of kind Span even though each segment also carries a categorical label.

B.6 Task specialization

The signature determines the semantic construction, but does not identify every benchmark-specific task. Let $P(\tau)$ denote constraints on the admissible inputs, conditioning, output, and temporal support of a task, excluding the orthogonal interface, rationale, protocol, and scoring fields.

We define

Table 6: Representative output types. †Requires model or human judgment and is therefore not directly usable as a verifiable reward.

Type	Kind κ	Default μ
$\textsc{Label}_K$, $\textsc{MultiLabel}_K$	VALUE	accuracy, macro-F1
$\mathbb{R}^d$	VALUE	MAE, MAPE
TEXT	VALUE	exact match; model judge†
$\textsc{Spans}(\mathcal{T})$	SPAN	event-F1, IoU
$\textsc{Signal}(\mathcal{T}', c)$	SIGNAL	MASE, DTW, task-specific metrics
$\textsc{Index}_N$, $\textsc{Ranking}_N$	INDEX	accuracy, MRR, Recall@k
$\textsc{Dist}(O)$	$\kappa(O)$	CRPS, NLL, calibration
$\textsc{Set}(O_1 \times O_2)$	highest-precedence component	component-wise

$$\tau_a \preceq \tau_b \quad \Longleftrightarrow \quad \phi(\tau_a) = \phi(\tau_b) \ \wedge \ P(\tau_b) \subseteq P(\tau_a). \tag{24}$$

Thus τ_a specializes τ_b if both perform the same semantic operation and τ_a adds further restrictions. Examples include

$$\text{ECG rhythm classification} \preceq \text{classification}, \tag{25}$$

and

$$\text{anomaly localization} \preceq \text{temporal localization}. \tag{26}$$

Likewise, single-step forecasting specializes forecasting by restricting the target to one future time point, and probabilistic forecasting specializes it by constraining the target to a predictive distribution.

The stored `from_tasks` relation may record such derivations as provenance. It does not require the child Task to physically copy fields from its antecedent or itself enforce the partial order.

B.7 ORTHOGONAL AXES: INTERFACE, REASONING, AND PROTOCOL

The fields ι, ε, and π do not alter $\phi(\tau)$.

Interaction interface. Question answering is an interface rather than a semantic family. For example,

$$\text{“What state is present?”} \mapsto (\textsc{Value}, \textsc{Obs}, 1), \tag{27}$$
$$\text{“When does the event occur?”} \mapsto (\textsc{Span}, \textsc{Obs}, 1), \tag{28}$$
$$\text{“What happens next?”} \mapsto (*, \textsc{Fut}, 1), \tag{29}$$
$$\text{“Which event occurred first?”} \mapsto (*, *, \geq 2). \tag{30}$$

The same semantic task can therefore be exposed through a structured output, question, or instruction without changing its family.

Reasoning. For a base task

$$f(X, C) \to Y, \tag{31}$$

a rationale requirement may additionally request

$$f(X, C) \to (Y, E), \tag{32}$$

where E contains evidence, intermediate conclusions, or an explanation. This does not change the semantic signature of the underlying task.

Evidence can itself be constrained, for example to stored spans or annotations, allowing explanation quality and target correctness to be evaluated separately.

Protocol. Under $\pi =$ streaming, predictions are made at times $t_1 < t_2 < \cdots$, with each prediction receiving only the input prefix available at that point. Online anomaly detection and early classification are therefore protocol variants of localization and attribute inference rather than separate semantic families.

B.8 A COMPOSITIONAL TASK ALGEBRA

Conceptually, a task factorizes as

$$\text{Task} = \underbrace{\phi(\tau)}_{\text{semantic construction}} \times \underbrace{P(\tau)}_{\text{specialization}} \times \underbrace{\iota \times \varepsilon \times \pi}_{\text{presentation and protocol}} \times \underbrace{\mu}_{\text{scoring}}. \tag{33}$$

The signature determines the semantic operation, $P(\tau)$ determines the task specialization, the orthogonal axes determine how the task is exposed and supervised, and μ determines how predictions are evaluated.

B.9 TASK CONSTRUCTORS

The following constructors describe common transformations between task specifications. They formalize relationships between tasks rather than requiring corresponding public SDK operations.

Definition 3 (Task constructors)**.** Given a valid task specification τ:

- ASK$_q$ changes the interface to a natural-language question;
- EXPLAIN adds a rationale requirement;
- STREAM changes the access protocol to streaming;
- SHIFT$_h$ withholds the final h units of target-channel support from the input, moving the target to $\Delta =$ FUT;
- MASK$_M$ removes cells M from the input and uses them as targets, producing GAP or CHAN support;
- REDUCE$_g$ maps a structured target to a value, e.g., existence, count, duration, or summary statistics;
- PAIR$_\rho$ combines temporal objects through a relation ρ, producing a relational task; and
- RESTRICT introduces additional constraints while preserving the signature, i.e., specialization under $\preceq$.

Thus SHIFT, MASK, REDUCE, and PAIR may move a task between regions of the signature space; RESTRICT moves within a region; and ASK, EXPLAIN, and STREAM alter only orthogonal axes.

For example, a language-conditioned anomaly-localization task may be written conceptually as

$$\text{ASK}\big(\text{RESTRICT}(\texttt{temporal localization})\big), \tag{34}$$

while forecasting with an explanation is

$$\text{EXPLAIN}(\texttt{forecasting}). \tag{35}$$

B.10 CONSTRUCTOR PRECONDITIONS

Constructors are only valid when the supervision required by the resulting task exists.

Constructor	Effect	Precondition
ASK_q	$\iota \leftarrow$ question	The question determines the intended target unambiguously.
EXPLAIN	$\varepsilon \leftarrow$ text/evidence	A rationale target exists in stored annotations, or the rationale is evaluation-only.
STREAM	$\pi \leftarrow$ streaming	The task and metric are defined for predictions from prefixes.
SHIFT_h	$\Delta \leftarrow$ FUT	The target has temporal support and sufficient target-channel history remains before the prediction boundary.
MASK_M	$\Delta \leftarrow$ GAP/CHAN	M contains cells for which ground truth is stored.
REDUCE_g	$\kappa \leftarrow$ VALUE	g is deterministic and its codomain is registered in $\mathcal{O}$.
PAIR_ρ	$\alpha \leftarrow \geq 2$	ρ is defined for the paired target types.
RESTRICT	$\preceq$	Each restricted field is a subset of the corresponding parent constraint and the resulting task has at least one valid instance.

Type validity does not imply semantic usefulness. For example, mechanically shifting a record-level diagnosis does not necessarily define a meaningful prediction problem. Derived tasks should therefore retain their derivation as provenance, and task constructors must satisfy domain-specific as well as type-level validity conditions.

Proposition 2 (Type closure)**.** *If τ is valid and a constructor's preconditions hold, the constructed task is also type-valid.*

Consequently, a registry may expose many task views over the same underlying recordings. This growth does not require duplicating the source signals and, when the required supervision is already stored, does not require additional labeling.

B.11 SCOPE AND LIMITATIONS

The current specification covers supervised and self-supervised mappings from temporal views to typed targets. For example, masked reconstruction and next-step prediction can be represented through MASK and SHIFT applied to signal targets.

The following settings are not yet first-class:

1. *Interventional and counterfactual tasks.* Conditioning may describe an intervention, but the current support relation does not distinguish observational from interventional targets.
2. *Sequential decision-making.* Tasks whose outputs are actions that change subsequent observations require an interactive environment rather than a fixed stored target.
3. *Multi-turn and agentic interaction.* Dialogue over a record, tool use, and iterative observation/action loops are outside the current task specification.
4. *Structured outputs beyond the registry.* Examples include causal graphs over channels or hierarchical event structures. Such targets can be incorporated by registering new output types, but are not currently built in.

The specification is therefore intended to be extensible rather than exhaustive: new output types, relations, and task constraints can be added without changing the underlying representation of temporal recordings.

ACRONYMS

API application programming interface. 3–5, 15

CoT chain-of-thought. 8, 9

ECG Electrocardiogram. 1, 2, 6

ECG-QA Electrocardiogram Question Answering. 8, 9

EDF European Data Format. 8, 9

F1 F1 score. 8, 9

JSON JavaScript Object Notation. 5

MMD multi-domain multimodal dataset. 3

QA question answering. 2

SDK software development kit. 3

SQL Structured Query Language. 5

TFM Temporal Foundation Model. 1, 2, 4, 9

TSB-UAD time-series benchmark for univariate anomaly detection. 3

TSFM Time Series Foundation Model. 1, 3

TSLM Time Series Language Model. 1, 2

UCR University of California, Riverside. 3

UEA University of East Anglia. 3

UTSD Unified Time Series Dataset. 3