TS-HAYSTACK: A MULTI-TASK RETRIEVAL BENCHMARK FOR LONG-CONTEXT PHYSIOLOGICAL TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Time Series Language Models (TSLMs) promise natural-language question answering over real-world temporal data, but their ability to retrieve and relate events in long time-series remains largely untested. We introduce `TS-Haystack`, a needle-in-a-haystack benchmark of ten event-grounded question-answering tasks over long physiological recordings, with contexts from 100 seconds to 9 hours, spanning direct retrieval, temporal understanding, multi-step retrieval, and contextual anomaly detection. Using this benchmark, we find that existing TSLMs exhibit performance degradation on long-context tasks: accuracy declines with context length, models that directly tokenize time-series without compression run out of memory at every context length, and other TSLMs collapse toward near-zero accuracy in interval retrieval tasks when increasing time-series length. These findings align with existing literature on text and multi-modal long context retrieval. We further contextualize TSLM performances with an oracle baseline and signal-blind baseline, and isolate their failure modes with an Agentic Retrieval for Time Series (ARTS) approach, a Large Language Model (LLM) agent with signal-specific classifiers as tools. Our findings uncover new failure modes in TSLMs at long context and show potential for architectures that further decouple local signal perception from global retrieval. Our code and datasets will be made publicly available upon acceptance.

1 INTRODUCTION

Time-series analysis enables the extraction of temporal dependencies and meaningful patterns in domains such as finance, healthcare, climate modeling, and industrial IoT (Benidis et al., 2022). Various real-world applications and the increasing availability and frequency of data require the processing of long time-series horizons. Continuous physiological ICU recordings (Yèche et al., 2021), full-night polysomnography (Ghassemi et al., 2018), ambulatory ECG (Petrutiu et al., 2007), and wearable activity monitoring (Chan et al., 2024) routinely span eight to twenty-four hours, while the clinically relevant events within them last seconds to minutes. Such real-world applications require *retrieving*, *aggregating* and *relating* events across large amounts of contextualized data. Methodologically, the field has evolved from classical statistical models to deep learning architectures, which capture nonlinear patterns in high-dimensional data (Torres et al., 2021). Building upon this progression, time-series foundation models (TSFMs) and time-series language models (TSLMs) have emerged as a promising frontier for general time-series forecasting and reasoning. Similar to large language models (LLMs) for text and vision language models (VLMs) for images, TSLMs aim to achieve robust generalization across diverse datasets and temporal tasks (Zhang et al., 2024; Abdullahi et al., 2025; Ansari et al., 2025). TSLM architectures can differ in how they encode time-series signals into an LLM backbone. *Full resolution* approaches directly tokenize time-series signals and include them as input for the LLM. Such methods enable more fine-grained temporal representations using self-attention over all time-series tokens but incur high computational costs at longer sequence lengths due to the quadratic cost of attention with respect to time-series length (Wang et al., 2025a; Vaswani et al., 2017). Conversely, *fixed-latent* methods compress the signal representation into a fixed set of tokens for the LLM. These methods improve efficiency through compression, but their ability to preserve temporal granularity at long context remains an open question (Jaegle et al., 2021;

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.

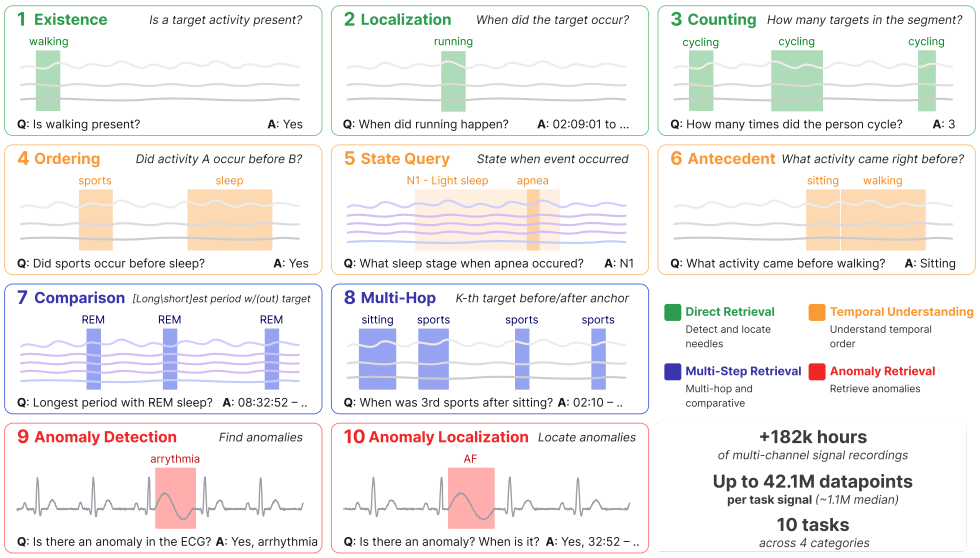


Figure 1: **TS-Haystack Tasks Overview.** Ten tasks spanning four categories for time-series retrieval and understanding. Tasks are instantiated with sample lengths ranging from 100 s up to 9 h (with average length 1 h 14 min) across more than 180k hours of generated benchmark multi-channel recording tasks (up to 13 channels). A view of channel count, frequency and lengths per dataset is provided in Table 1.

Wang et al., 2025c). Several established benchmarks have been proposed to understand the behavior of LLMs on long-context retrieval tasks for text (Kuratov et al., 2024; Kamradt, 2023), vision (Wang et al., 2025b), and audio (He et al., 2025), unveiling performance degradation at increased context lengths, and leading to architectural breakthroughs in long context retrieval (Zhang et al., 2025; Behrouz et al., 2025; Gu & Dao, 2023). Existing benchmarks for TSLMs have explored statistical reasoning (Cai et al., 2024), short-context classification (Dau et al., 2019; Bagnall et al., 2018), and chain-of-thought augmented classification (Langer et al., 2025). However, determining whether these models can localize, count, or compare events at longer temporal horizons remains unexplored, and a standardized benchmark for long-context retrieval and understanding is missing for time-series.

We present **TS-Haystack**, a novel benchmark for event retrieval and temporal understanding over long physiological time-series that enables systematic diagnosis of long-context retrieval in TSLMs. It comprises ten tasks across four categories: direct retrieval, temporal understanding, multi-step retrieval, and anomaly retrieval, as shown in Figure 1. Each task is instantiated at context lengths from 100 s up to 9 h at sampling rates of 100 Hz and 128 Hz, enabling the study of retrieval and understanding ability relative to the temporal scale. To create this benchmark, we extend the needle-in-a-haystack paradigm (Hsieh et al., 2024) to time-series, enabling generation of retrieval-based question-answer pairs from existing long horizon labeled time-series data. We create samples under two complementary protocols. When local signals are uninformed by their surrounding context (e.g., local activity recognition), we synthesize samples by embedding existing real annotated events (*needles*) into other longer background recordings (*haystacks*). For domains where local events are hard to disentangle from their local context (e.g., heartbeat anomalies, sleep stages that follow a cyclic macro-trend), we extract naturally occurring segments (e.g., a contiguous 1-hour window of an ECG recording with its native beat annotations, a 2-hour slice of polysomnography with the original sleep-stage and arousal labels) from the source data and formulate questions about each extracted sample. We detail the trade-offs, provide a framework and statistically validate needle insertion in our dataset generation pipeline (Section 3.2). We evaluate common TSLMs such as OpenTSLM (Langer et al., 2025), ITFormer (Wang et al., 2025c) and ChatTime (Wang et al., 2025a) on `TS-Haystack`. Our results uncover new architecture gaps and failure modes on long-context retrieval. For comparison, we introduce a series of baselines: First, we render each recording's ground-truth annotations as a list of events and ask an *oracle* model to answer questions from that list. This bypasses local perception and confirms the LLM backbone's reasoning capabilities. Second, we fine-tune a *signal-blind* LLM exclusively on the benchmark's question-answer pairs while withholding the signal.

This baseline measures how well LLMs without access to the time-series perform on label priors and question phrasing alone. Third, we introduce Agentic Retrieval for Time-Series (ARTS), a baseline that uses an LLM orchestrator and signal-specific classifiers to decouple local perception from global retrieval. We find that TSLM architectures that ingest time-series tokens without fixed-latent compression run out of memory at every context length. For other TSLMs, performance degrades with context length to near-zero beyond lengths of 1 h for localization-specific tasks, and fallback to accuracies similar to those of the signal-blind LLM on other tasks (see Appendix A.1). Together, ARTS and the oracle baseline suggest that preserving fine-grained perception may be a bottleneck in TSLM performance. `TS-Haystack` acts as a diagnostic tool for TSLM architectures in long context retrieval and motivates future work in architectures decoupling local perception and global reasoning.

2 Related Work

In the following, we present related work on Time-series Language Models (TSLMs) as well as their different encoding mechanisms for time series. We also present existing time-series benchmarks for classification, retrieval and forecasting, and needle-in-a-haystack benchmarks in other modalities.

Time-Series Language Models TSLMs are multi-modal LLMs that integrate time-series as a native modality (Langer et al., 2025). We categorize different architectures by how they map the time-series input of length L into the language model's input space. *Full-resolution adapters*, such as ChatTime (Wang et al., 2025a) and ChatTS (Xie et al., 2025), pass time-series tokens or $N = L/P$ patch embeddings directly to the LLM, preserving signal fidelity at the cost of $\mathcal{O}(N^2)$ attention in the extended LLM context, which limits scaling to long contexts. ChatTime tokenizes the signal directly (Wang et al., 2025a), while ChatTS adopts a LLaVA-style (Liu et al., 2023) insertion of patch embeddings into the LLM input. *Fixed-latent* methods instead integrate a module between the encoder and the LLM, which learns to compress the time series. ITFormer (Wang et al., 2025c) aggregates patch embeddings into a fixed set of summary tokens, while OpenTSLM-Flamingo (Langer et al., 2025) adopts a Perceiver Resampler (Jaegle et al., 2021) in which M learned queries $\mathbf{Q} \in \mathbb{R}^{M \times d}$ cross-attend (Vaswani et al., 2017; Alayrac et al., 2022) to the N patch embeddings and produce an output $\mathbf{Z} \in \mathbb{R}^{M \times d}$ of constant size, enabling linear scaling in L.

Time-Series Question-Answering Benchmarks Several benchmarks for time-series evaluate fixed-length categorization (Dau et al., 2019; Bagnall et al., 2018) or forecasting (Li et al., 2025; Aksu et al., 2024), yet do not test natural-language retrieval and understanding over time-series in the ways that text and vision benchmarks have explored (Yue et al., 2024; Singh et al., 2019). Recent TSLM-centric benchmarks include text and time-series data, but remain restricted in scope: TimeSeriesExam (Cai et al., 2024), HEARTS (Li et al., 2026) or ECG-QA (Oh et al., 2023) probe statistical understanding or semantic time-series classification via multiple-choice questions on short series. Due to this gap, TSLM implementations have focused their evaluations on short-context classification with optional chain-of-thought rationale generation (Langer et al., 2025; Wang et al., 2025c), leaving an open question on their ability to retrieve and relate events at long horizons.

Long-Context Retrieval Benchmarks The needle-in-a-haystack (NIAH) paradigm probes retrieval by embedding target events (*needles*) within a long background of unrelated content (the *haystack*) (Kamradt, 2023; Kuratov et al., 2024), requiring the model to recover these needles to answer direct or compositional questions. NIAH benchmarks have revealed reasoning and retrieval degradation of LLMs on text (Liu et al., 2024), vision (Wang et al., 2024a; Dingjie et al., 2024) and audio (He et al., 2025). Extending this paradigm to time-series is non-trivial: unlike text, where a needle sentence can be cleanly inserted into an unrelated passage, time-series signals lack discrete, semantically separable units. Naive insertion for time-series risks two failure modes: the introduction of artifacts at the seam that may trivialize retrieval through pattern matching and the destruction of context-dependent semantics for labeled events (e.g., a segment of ECG is labeled as arrhythmia yet its diagnosis depends on adjacent beats excluded from the labeled event). We address this in Section 3.2.

Agentic Long-Context Reasoning Another line of work studies *agentic* frameworks in which an LLM iteratively issues tool calls and integrates intermediate results into its context (Yao et al., 2023). Such methods have been shown to recover much of the long-context performance lost by single-pass question answering, and have been applied to long-form multimodal retrieval (Wang et al., 2024b).

3 METHODS: TS-HAYSTACK

We propose TS-Haystack, a benchmark that repurposes long, annotated physiological recordings from three modalities into ten retrieval and understanding tasks instantiated at context lengths from 100 s up to 9 h. We first formalize long context retrieval in time-series (Section 3.1), and present the source datasets, task categories and dataset construction details of TS-Haystack (Section 3.2). Later, we motivate semi-synthetic and natural-segment sample construction methods to turn annotated time-series corpora into needle-in-a-haystack (NIAH) retrieval benchmarks for time-series (Section 3.2).

3.1 TSLMS AND NIAH FORMALIZATION FOR TIME-SERIES

TS-Haystack adapts the needle-in-a-haystack paradigm to time-series, casting long-context event retrieval as a question-answering problem over multi-channel recordings. Each TS-Haystack sample consists of three components: a long recording (the *haystack*), a set of sparse annotated *events of interest* embedded within it (the *needles*), and a natural-language question grounded in those needles. Task difficulty arises because needles occupy a vanishing fraction of the total recording length: a successful model must retrieve, localize, and relate events that are sparse in time, preserving the fine-grained signal detail required to identify them. We formalize this as follows. Let $\mathbf{X} \in \mathbb{R}^{L \times C}$ denote a multi-channel recording of length L with C channels, and let q denote a natural-language question over $\mathbf{X}$. Each TS-Haystack sample is a tuple $(\mathbf{X}, q, y)$, where y is a structured answer (boolean, integer, label, or interval). The annotation $\mathcal{A}(\mathbf{X}) = \{(s_i, e_i, c_i)\}_{i=1}^{n}$ comprises n disjoint events of interest, each defined by start and end indices $0 \leq s_i < e_i \leq L$ and a class label $c_i \in \mathcal{C}$ drawn from the source dataset's annotations. By construction, the aggregate needle mass satisfies $\sum_i (e_i - s_i) \ll L$. We define a TSLM as a function $f_\theta : (\mathbf{X}, q) \mapsto \hat{y}$ trained to maximize task-specific accuracy. We hypothesize that the constant-size compression employed by fixed-latent architectures introduces an information bottleneck under long contexts: as L grows, the linearly increasing compression ratio N/M forces each latent token to progressively marginalize away information relevant to q. For the three TSLM methods described in Section 2, OpenTSLM-Flamingo and ITFormer compress the time series into a fixed set of latent tokens, while ChatTime passes time-series tokens directly to the LLM. In the example of OpenTSLM-Flamingo, M defaults to 64, yielding a $\sim$2,800$\times$ compression ratio at 2 h of context sampled at 100 Hz with patch size 4. TS-Haystack is designed to stress both encoding regimes by the formalized needle structure above, where the answer depends on intervals whose aggregate length becomes negligible relative to L.

3.2 DATASET CONSTRUCTION

TS-Haystack draws from three openly available, annotated long-form physiological datasets selected for diversity in modality, sampling rate, and event sparsity (Table 1): Capture24 (Chan et al., 2024) (wrist accelerometer), Sleep Polysomnography (PSG) (Ghassemi et al., 2018), and LTAF (Long-Term Atrial Fibrillation) (Petrutiu et al., 2007). Although these domain-specific datasets provide long, expert-annotated recordings, they are released with per-window classification labels and have not been formulated as retrieval or question-answering tasks. To address this, we draw existing labeled segments (*needles*) from the source recordings and turn them into retrieval targets for TSLMs, presenting a suite of retrieval tasks formulated as question–answer pairs using two complementary sample creation protocols motivated by the specific constraints of time-series signals (see Section 3.2).

We instantiate TS-Haystack by assigning recordings to a train/validation/test split per dataset and organize tasks around relevant questions for long, continuous physiological recordings, grouped into four categories. *Direct Retrieval* (Existence, Localization, Counting) probes whether models maintain temporal granularity at scale; *Temporal Understanding* (Ordering, State Query, Antecedent) tests understanding of order and co-occurrence; *Multi-Step Retrieval* (Comparison, Multi-Hop) requires composing multiple retrievals; and *Anomaly Retrieval* (Anomaly Detection, Anomaly Localization) evaluates detection and localization of relevant anomalies. Task categories are motivated by needle-in-a-haystack benchmarks in text (Hsieh et al., 2024; Yu et al., 2025) and vision (Wang et al., 2024a), where direct single-needle retrieval is tested along with multi-needle tasks including comparison and counting. We include additional tasks uniquely relevant to time-series such as Anomaly detection and segment localization (Oh et al., 2023; Cai et al., 2024). Figure 1 presents a visualization of the ten tasks, answer types and example questions. Each task is implemented using a family of paraphrased

Table 1: TS-Haystack source datasets and sample construction method. Sleep PSG is downsampled from 200 Hz. *Classes* reports the size of each label vocabulary (5 AASM stages + 5 arousal classes for Sleep; 4 beat classes + 9 rhythm codes for LTAF). Further statistics appear in Appendix B.1.

Dataset	Modality	Channels	Freq. (Hz)	Classes	Context lengths	Sample construction
Capture24[a]	Wrist accelerometer	3	100	10	100 s, 15 min, 1 h, 2 h	Semi-synthetic
Sleep PSG[b]	Polysomnography	13	100	10	100 s, 15 min, 1 h, 2 h, full night[†]	Natural
LTAF[c]	2-lead ECG	2	128	13	100 s, 15 min, 1 h, 2 h	Natural

[a]Chan et al. (2024); [b]Ghassemi et al. (2018); [c]Petrutiu et al. (2007). [†] Full recording: median 7.74 h; maximum 9.5 h.

natural-language templates and admits a deterministic ground-truth answer computed from the underlying annotations. Complete per-dataset details and statistics are reported in Appendix B.

Needle-in-a-haystack sample creation for time-series We propose two protocols to turn existing annotated time-series samples into needle-in-a-haystack (NIAH) retrieval questions for TSLMs. The first is *semi-synthetic needle insertion*. In this protocol, we sample annotated events (*needles*) in the time-series from their original context and insert them back into a different background (*haystack*) recording (e.g. a 10 s "Running" event from Participant A placed into a 1 h "Sitting"-only background from Participant B). The second is *natural segment sampling*. Here, we select a fixed-length slice of a recording, and a retrieval question is asked about a naturally occurring event in the signal (e.g. *How many apnea events did the patient suffer?* for a 2 h slice of a full night of polysomnography recording). In the following, we motivate the choice between the two methods, detail the insertion protocol, and propose a framework to determine the appropriate use of each. Table 1 summarizes the sample generation choice for each dataset in TS-Haystack and the trade-offs.

Semi-synthetic needle insertion Semi-synthetic needle insertion is motivated by two limiting factors present in natural long-horizon recordings: class imbalance and predictable label transitions. As samples become longer, signal events closely mirror the underlying structure of the physical processes recorded by these time-series, such as participants predictably transitioning from "Sitting" to "Standing" in activity labels or a "Light sleep (N1)" cycle transitioning to "Deeper light sleep (N2)" in a sleep recording. To address this limitation, we draw inspiration from NIAH benchmarks in text (Yu et al., 2025; Hsieh et al., 2024), where small fragments of text (needles) are programmatically inserted in a large corpus of unrelated text (haystacks). We adapt this to time-series, drawing real annotated signal samples and inserting them back into existing recordings matching programmatically stated requirements (e.g. a background without pre-existing "Running" labels). Formally, given a target context length L and a queried class c, we sample a background recording from a participant, select one or more annotated bouts $\mathbf{x}_{\text{needle}}$ of class c from a random participant in the same split, and insert them at controlled locations replacing the original segment. The number, location, and labels of these needles determine the answer key for each task. Semi-synthetic needle insertion allows for controlled difficulty modulation (needle length, count, and spacing) and augmented sample generation (recombination of needles and backgrounds beyond the corpus's natural yield).

Insertion biases and natural segment sampling Naive semi-synthetic insertion may introduce biases specific to the time-series domain, which we address below. First, semi-synthetic sample construction might produce insertion artifacts that models could exploit to answer the question. To test whether our insertion procedure introduces detectable local seam artifacts, we train an XGBoost (Chen & Guestrin, 2016) classifier to distinguish background-only windows from windows with a same-activity needle inserted from a different participant. The same local blending operation is used at every benchmark context length. An AUC ≈ 0.50 indicates that the insertion produces no systematic detectable artifact under the tested setting (Appendix C.6). Additionally, labeled events may not being interpretable without their surrounding context. In the Sleep domain, sleep stages are scored over 30 s epochs embedded in the cyclic NREM/REM macrostructure encoded by AASM rules (Berry et al., 2017); in ECG, signals carry strong inter-beat dependencies (R-R autocorrelation, beat-to-beat morphology). Thus, slicing signals based on the annotation span alone might alter the semantic meaning of the label. We consider a signal as *locally context-independent* for class c if the distribution of a class-c bout conditional on its surrounding context is well-approximated by its marginal distribution

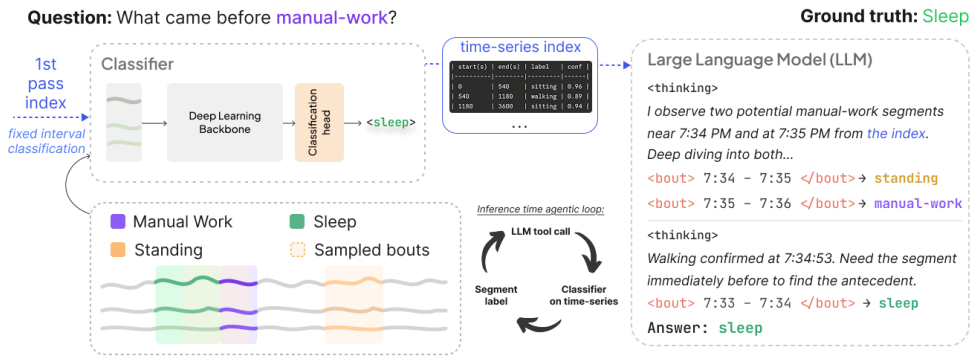


Figure 2: **ARTS.** A classifier pre-pass sweeps the recording over non-overlapping tiles, producing a textual timeline of predicted labels and confidences for the orchestrator as initial context. The orchestrator issues `<bout>` tool calls to re-classify specific segments and reason toward a final answer.

given the label, $p(\mathbf{x}_{[s:e]} \mid \mathbf{X}_{\setminus[s:e]}, c) \approx p(\mathbf{x}_{[s:e]} \mid c)$ (Appendix B.2). If this holds true, transplanting a bout into a different background can be considered to preserve its local semantics, and insertion yields composites statistically close to naturally observed recordings. The resulting signal remains locally coherent in meaning, while succeeding in breaking predictable label transitions. Activity recognition is conventionally cast as classification over short, locally segmented windows (Bulling et al., 2014): a bout of running carries the same activity label whether it follows sitting or walking. Sleep and ECG do not satisfy the condition, as labeled events are not naturally interpretable without their surrounding context (Berry et al., 2017). As such, we use semi-synthetic sample construction to generate NIAH retrieval tasks for wrist accelerometer signals (Capture24), and natural segment sampling for Sleep Polysomnography (PSG) and LTAF (Long-Term Atrial Fibrillation) ECG, as shown in Table 1.

4 Experiments

In the following, we present the different baselines that help contextualize and isolate failure modes beyond TSLM results, as well as TS-Haystack task instantiation and TSLM training details.

4.1 Baselines: Decoupling Perception from Retrieval

Existing TSLM architectures couple local perception and global retrieval into a single forward pass. To isolate the failure modes of TSLMs in long-context retrieval, we introduce an agentic retrieval for time-series (ARTS) baseline that decouples *local perception* (short segments at full resolution) from *global retrieval* (locating and relating events in a compact symbolic view of the whole recording). ARTS reframes long-context retrieval as a two-stage process using an *orchestrator* f_θ, responsible for aggregating classified signal labels and answering the final question, and specialized *classifier tools* g_ϕ, responsible for fine-grained signal understanding over shorter segments of the full recording (Figure 2). In the first stage, a *classifier pre-pass* sweeps g_ϕ over a non-overlapping tiling of the full recording at a fixed window size, producing a textual timeline of predicted labels that summarizes $\mathbf{X}$ for the orchestrator. The orchestrator reads this *Time-series index* and iteratively issues `<bout>` tool calls to "zoom-in", re-classify selected segments and resolve ambiguities left by the pre-pass, integrating intermediate observations until it emits a final answer. Crucially, the orchestrator never ingests $\mathbf{X}$ directly: it operates entirely over the symbolic timeline produced by g_ϕ, decoupling local perception from global retrieval. The pre-pass amortizes the cost of covering the full recording into a single sweep, allowing the agentic loop to focus on targeted refinement rather than blind exploration. The orchestrator is an off-the-shelf LLM used zero-shot. We instantiate it with a range of seven open-weight and proprietary base LLMs of different sizes: Qwen3.5-2B (Qwen, 2026), Qwen3.5-9B (Qwen, 2026), Qwen3.5-35B-A3B (Qwen, 2026), gpt-oss-120b (OpenAI, 2025), DeepSeek-V3.2 (DeepSeek-AI, 2025), Gemini 3.8 Flash (Google DeepMind, 2026), and GPT-5.4 (Singh et al., 2025). Every orchestrator is used at the provider's default sampling temperature, with the classifier tools, pre-pass and prompts held fixed. The classifier tool g_ϕ is instantiated per domain: for Capture24, a dual-encoder model concatenating frozen Chronos-2 (Ansari et al., 2025) and OxWearables (Yuan et al.,

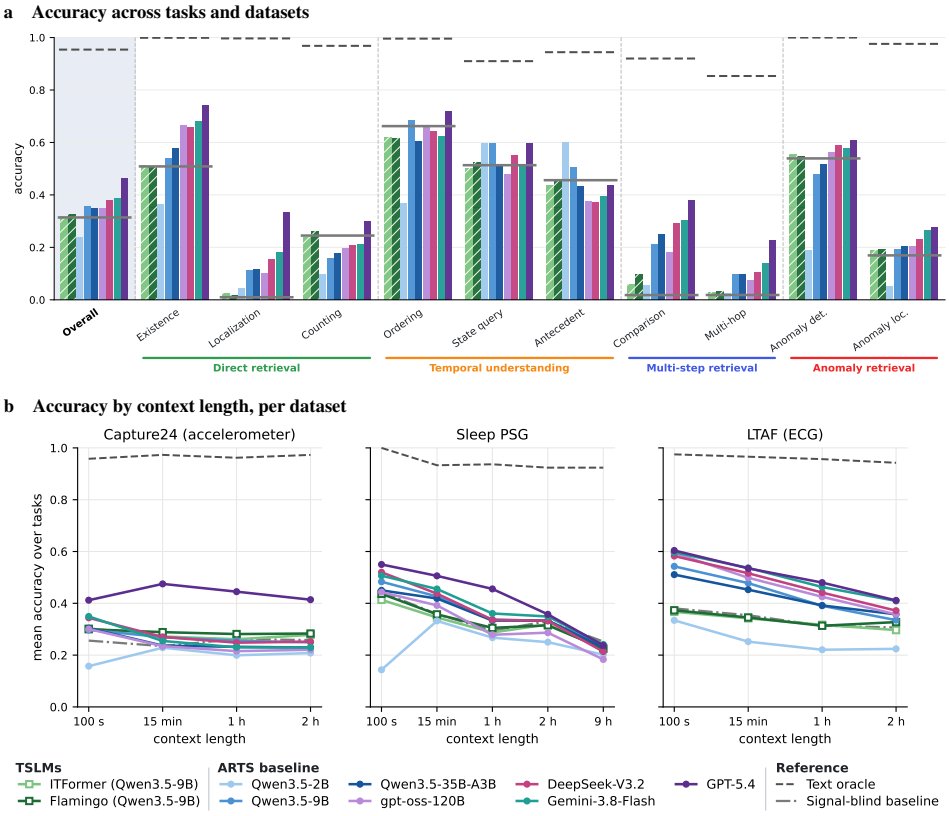


Figure 3: **Accuracy against the signal-blind baseline and across context lengths.** Figure **a**: accuracy per task, averaged over the three datasets and all context lengths. Figure **b**: the same accuracies resolved by context length, one panel per dataset.

2024) embeddings, with a two-layer multi-layer perceptron (MLP) classification head trained on Capture24 for activity recognition. For Sleep, we use two parallel Chronos-2 heads for sleep staging and arousal detection. For ECG, we use two complementary classifiers trained from scratch on LTAF: a history-time-frequency CNN ensemble with a per-class MLP head for per-beat classification, and a wide 1D-ResNet with test-time augmentation for per-rhythm classification (Wang et al., 2021; Li et al., 2022; Hannun et al., 2019). By decoupling orchestration from local perception, ARTS leverages existing domain-specific segment classifiers and off-the-shelf LLMs, enabling the use of validated classifiers for question answering. A detailed explanation of their construction details can be found in Appendix Sections C.7, C.9 and C.11.

Upper and lower-bound performance Two additional reference baselines contextualize accuracy. The *oracle* bypasses perception entirely, receiving the ground-truth annotation $\mathcal{A}(\mathbf{X})$ as a textual list of timestamped events and answering q from it. Its accuracy measures how well the chosen LLM answers from ground-truth annotations, and quantifies the gap induced by a model's perception-induced loss. A second *signal-blind* baseline is introduced to quantify a fine-tuned model's ability to recover the answer without signal access. This model never sees the signal, and is exclusively fine-tuned on the question–answer pairs. Its accuracy measures how much of a question is answerable from label priors and phrasing alone; a finetuned TSLM at or below it has not demonstrably used the signal. The oracle is instantiated with GPT-5.4 over the textual annotation, and the signal-blind baseline with the same Qwen3.5-9B backbone used by the TSLMs, fine-tuned on the question–answer pairs of each dataset. Appendix C.4 presents further details on these baselines.

TS-Haystack training and evaluation We generate train, validation, and test samples for every instantiated cell (spanning task, context length, source dataset). Existence and Anomaly Detection are not instantiated at longer lengths for Sleep and ECG, since the held-out target classes are almost

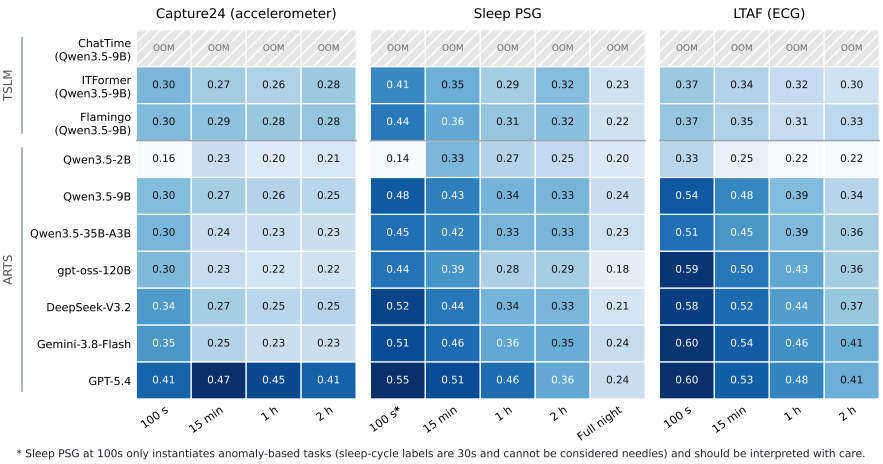


Figure 4: **Model accuracy by dataset and context length.** Each cell gives the mean accuracy over tasks at that context length. ChatTime exhausts GPU memory at every context length and is marked OOM throughout.

certainly present in the natural recordings at such lengths. Per-cell counts and the complete split tables are reported in Appendix B.1. We train three TSLM architectures: OpenTSLM-Flamingo (Langer et al., 2025), ChatTime (Wang et al., 2025a) and ITFormer (Wang et al., 2025c). All architectures use Qwen3.5-9B (Qwen, 2026). Models are trained jointly across all tasks and context lengths per signal domain. Training hyperparameters are reported in Appendix C.1. Discrete-answer tasks (existence, counting, ordering, state query, antecedent, anomaly detection) are scored by exact match after type-aware normalization. Time range tasks (localization, comparison, multi-hop, anomaly localization) are scored as correct if the predicted interval has IoU ≥ 0.25 with the ground truth.

4.2 Results

Per-dataset accuracy and drift We evaluate Open-TSLM Flamingo, ITFormer, ChatTime and ARTS with seven orchestrators across every (task, context length, dataset) cell of TS-Haystack; per-task results underlying these aggregates are reported in Appendix A. Figure 3a plots every model's accuracy contextualized by the signal-blind and oracle metrics of each task, Figure 3b displays the same accuracies by context length, and Figure 4 shows mean accuracy per model, context length and dataset. ChatTime using a Qwen 3.5 9B backbone runs out of memory at every context length and dataset and is reported as such. At the longest evaluated context in each dataset, ITFormer and Flamingo achieve Ordering accuracies of $0.50/0.52$ on Capture24, $0.61/0.55$ on Sleep, and $0.75/0.74$ on ECG. The corresponding signal-blind accuracies are 0.55, 0.75, and 0.79. Meanwhile, on Localization at these same horizons, both TSLMs score 0.01 on Capture24 and 0.00 on Sleep and ECG. This task-relative performance flips under the ARTS baseline. On Localization, GPT-5.4 ARTS scores 0.55 on Capture24, 0.26 on ECG, and 0.06 on Sleep, despite scoring lower than TSLMs on Ordering and Antecedent tasks at those context lengths. Figure 5 displays the task-relative context degradation per $\log_{10}$ length increase. Accuracy falls by 6.6 points on average, with a 95% interval of $[-9.5, -4.0]$ obtained by resampling tasks. TSLMs exhibit lower decay profiles relative to the initial length task performance, being closest to the signal-blind baseline.

5 Discussion

Context-length degradation in time-series mirrors other modalities Across all evaluated TSLMs, we observe an overall decline in accuracy with context length on TS-Haystack. The oracle's stability across the same range (0.93–0.97) suggests the benchmark is well constructed and answers are recoverable from the underlying annotations. This pattern is consistent with documented degradation in long-context LLMs for text (Liu et al., 2024), vision (Wang et al., 2024a; Dingjie et al., 2024), and audio (He et al., 2025), and to our knowledge constitutes its first systematic characterization for time-series.

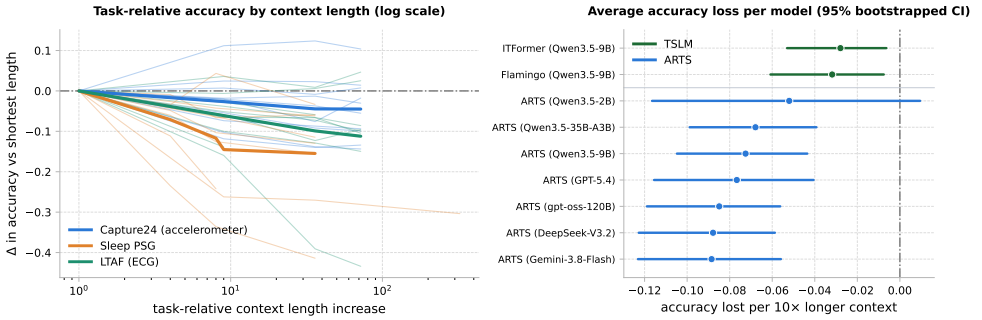


Figure 5: **Task-relative change in accuracy by context length increase.** Left: Accuracy drift relative to a task's shortest context. Bold lines show aggregate degradation per dataset. Right: Accuracy lost per unit increase in $\log_{10}$ context length, with a 95% interval obtained by resampling tasks. Construction details are given in Appendix C.3.

Architectural choices interact with task type, not just context length The per-task breakdowns (Appendix A) reveal further insights on the performance differences of TSLMs. Tasks with discrete answers retain performance across context lengths, while tasks with time-interval answers collapse toward zero by 1–2 h. The gap between models and the oracle is widest precisely on these localization tasks, suggesting the bottleneck at long context is fine-grained temporal grounding rather than high-level reasoning, and fine-tuning allows the model to maintain performance at the signal-blind baseline. ARTS achieves the highest accuracy in all time-series domains at their initial context lengths, leading the next-best baseline by 13–25 percentage points (pp) for the GPT-5.4 orchestrator and maintains relative performance improvement at most context lengths. Yet ARTS also exhibits the largest absolute performance degradation (-19 pp on ECG, -31 pp on Sleep). This decline concentrates on time-interval tasks (Localization, Comparison, Multi-hop, Anomaly Localization), where ARTS starts from a higher performance than TSLMs but degrades steeply. Conversely, TSLMs outperform ARTS on tasks such as State Query and Antecedent, especially in naturally-sampled datasets, which could point to their internalizing marginal label distributions in fine-tuning that ARTS' inference-only agentic loop cannot exploit (see Appendix A). The sustained accuracy of the oracle suggests ARTS' decline may arise from compounding classifier errors over longer traces, and TSLMs may be bottlenecked by temporal grounding rather than LLM aggregation capabilities.

Needle insertion for time-series presents domain specific constraints Time-series events may be governed by transition patterns of the real-world systems they represent. TS-Haystack adapts the needle-in-a-haystack paradigm to time-series with two distinct sample construction methods. Our adaptation to this new modality shows signal and dataset specific constraints. Annotated events may only maintain their meaning conditional on surrounding context, and signal insertion on a continuous data stream requires a validation pipeline particular to each domain. TS-Haystack presents the first systematic evaluation of TSLMs on a comprehensive set of long context needle-in-a-haystack retrieval tasks. An extensive contextualization of results using different baselines situates the bottleneck in fine-grained local perception and retrieval rather than LLM capabilities, granted a digestible view of the signal labels is provided to the model. Our results suggest existing TSLM architectures fallback to signal-blind question–answer learning while struggling to localize specific segments in long context.

Limitations and future work While our work presents a general framework for the generation of long-context retrieval benchmarks from any labeled time-series dataset, extending it to non-physiological domains such as industrial sensor streams (Saxena et al., 2008) and climate signals (Hersbach et al., 2020) remains future work. Additionally, our needle-insertion protocol is applied to accelerometer signals; extending it to additional domains where local-segment independence holds is a natural follow-up. Finally, ARTS is conceived as a promising yet naive approach for decoupling local perception from global retrieval: its degradation at long contexts motivates future work on further decoupling using exploratory sub-agents (Zhang et al., 2025) and methods for better signal classification boundaries in fine-grained perception.

Ethics statement

TS-Haystack is built exclusively from openly available, de-identified datasets released for research use under their respective licenses and data-use agreements: Capture24 (Chan et al., 2024), the PhysioNet/CinC 2018 sleep polysomnography dataset (Ghassemi et al., 2018), and the Long-Term Atrial Fibrillation database (Petrutiu et al., 2007). No new data were collected from human subjects, and no attempt was made to re-identify participants. The benchmark repurposes existing per-window annotations into question-answer pairs and does not introduce new labels about individuals. The evaluated methods, including the agentic framework, are diagnostic research artifacts and are not intended for clinical decision making; we caution against deploying any of the evaluated systems in medical or safety-critical settings without domain-specific validation. We do not foresee direct negative societal impacts beyond those generally associated with improved automated analysis of physiological sensor data, such as privacy concerns that we mitigate by using only public, de-identified sources.

Reproducibility statement

We have taken several steps to make our results reproducible. The formal problem setup and the two sample-construction protocols are described in Section **??**, with the formal context-independence condition and the needle-insertion protocol detailed in Appendix B.2 and its statistical validation in Appendix C.6. Complete per-dataset statistics, split tables and per-cell sample counts are provided in Appendix B.1, and per-task construction details (question templates and deterministic answer computation) in Appendix C.12. Training hyperparameters, optimizer settings and evaluation protocol for every TSLM are listed in Appendix C.1; the orchestrator configuration and per-domain classifier construction for ARTS are given in Appendix Sections C.7, C.9 and C.11; and the closed-form random baselines in Appendix **??**. Sample generation uses seeded random-number generators so that the benchmark can be regenerated deterministically from the public source datasets. The benchmark generation pipeline, evaluation code, and the generated datasets will be released publicly upon publication.

AI use statement

In this work, we used generative AI tools to assist with editing and polishing the manuscript text and with writing and debugging auxiliary code (e.g., data-processing scripts, plotting utilities, and LaTeX formatting). All AI-generated code has been reviewed by authors for correct methodology. We did not use generative AI tools for research ideation, for formulating the scientific claims, or for producing the experimental results. Separately, large language models are a component of the evaluated methods, TSLMs and ARTS orchestrator and the oracle baseline, as described in Sections 3–4; this constitutes the object of study rather than an authoring aid. We take responsibility for the final content of this work, including text, claims or artifacts.

References

Shamsu Abdullahi, Kamaluddeen Usman Danyaro, Abubakar Zakari, Izzatdin Abdul Aziz, Noor Amila Wan Abdullah Zawawi, and Shamsuddeen Adamu. Time-series large language models: A systematic review of state-of-the-art. *IEEE Access*, 13:30235–30261, 2025. doi: 10.1109/ACCESS.2025.3535782.

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 Preprints*, 2024. doi: 10.48550/arXiv.2410.10393.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp.

23716–23736, 2022. URL https://proceedings.neurips.cc/paper/2022/hash/960a172bc7fbf0177ccccbb411a7d800-Abstract-Conference.html.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv Preprints*, 2025. doi: 10.48550/arXiv.2510.15821.

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 Preprints*, 2018. doi: 10.48550/arXiv.1811.00075.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://neurips.cc/virtual/2025/loc/san-diego/poster/119639.

Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6):1–36, 2022. doi: 10.1145/3533382.

Richard B. Berry, Rita Brooks, Charlene Gamaldo, Susan M. Harding, Robin M. Lloyd, Stuart F. Quan, Matthew T. Troester, and Bradley V. Vaughn. *The AASM Manual for the Scoring of Sleep and Associated Events: Rules, Terminology and Technical Specifications, Version 2.4*. American Academy of Sleep Medicine, Darien, IL, 2017.

Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys*, 46(3):1–33, 2014. doi: 10.1145/2499621.

Yifu Cai, Arjun Choudhry, Xing Hu, and Artur Dubrawski. Timeseriesexam: A time series understanding exam. *arXiv Preprints*, 2024. doi: 10.48550/arXiv.2410.14752.

Shing Chan, Hang Yuan, Catherine Tong, Aidan Acquah, Abram Schonfeldt, Jonathan Gershuny, and Aiden Doherty. CAPTURE-24: A large dataset of wrist-worn activity tracker data collected in the wild for human activity recognition. *Scientific Data*, 11(1):1135, 2024. doi: 10.1038/s41597-024-03960-3.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016. doi: 10.1145/2939672.2939785.

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 Journal of Automatica Sinica*, 6(6):1293–1305, 2019. doi: 10.1109/JAS.2019.1911747.

DeepSeek-AI. Deepseek-v3.2: Pushing the frontier of open large language models, 2025. URL https://arxiv.org/abs/2512.02556.

Song Dingjie, Shunian Chen, Guiming Hardy Chen, Fei Yu, Xiang Wan, and Benyou Wang. Milebench: Benchmarking mllms in long context. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Uhwze2LEwq.

MM Ghassemi, B Moody, L Lehman, C Song, Q Li, H Sun, M Westover, and GD Clifford. You snooze, you win: the PhysioNet/computing in cardiology challenge 2018. In *2018 Computing in Cardiology Conference (CinC)*, pp. 1–4, 2018. doi: 10.22489/CinC.2018.049.

Google DeepMind. Gemini 3.8 Flash model card. https://deepmind.google/models/model-cards/gemini-3-8-flash/, September 2026. Published 2 September 2026. Accessed: 2026-09-14.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv Preprints*, 2023. doi: 10.48550/arXiv.2312.00752.

Awni Y. Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H. Tison, Codie Bourn, Mintu P. Turakhia, and Andrew Y. Ng. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65–69, 2019. doi: 10.1038/s41591-018-0268-3.

Peize He, Zichen Wen, Yubo Wang, Yuxuan Wang, Xiaoqian Liu, Jiajie Huang, Zehui Lei, Zhuangcheng Gu, Xiangqi Jin, Jiabing Yang, et al. Audiomarathon: A comprehensive benchmark for long-context audio understanding and efficiency in audio llms. *arXiv Preprints*, 2025. doi: 10.48550/arXiv.2510.07293.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, and J.-N Thépaut. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146, 06 2020. doi: 10.1002/qj.3803.

Gerhard Hindricks, Tatjana Potpara, Nikolaos Dagres, Elena Arbelo, Jeroen J. Bax, Carina Blomström-Lundqvist, Giuseppe Boriani, Manuel Castella, Gheorghe-Andrei Dan, Polychronis E. Dilaveris, Laurent Fauchier, Gerasimos Filippatos, Jonathan M. Kalman, Mark La Meir, Deirdre A. Lane, Jean-Pierre Lebeau, Maddalena Lettino, Gregory Y. H. Lip, Fausto J. Pinto, G. Neil Thomas, Marco Valgimigli, Isabelle C. Van Gelder, Bart P. Van Putte, and Caroline L. Watkins. 2020 ESC guidelines for the diagnosis and management of atrial fibrillation developed in collaboration with the European Association for Cardio-Thoracic Surgery (EACTS). *European Heart Journal*, 42(5): 373–498, 2021. doi: 10.1093/eurheartj/ehaa612.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models?, 2024. URL https://arxiv.org/abs/2404.06654.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning (ICML)*, pp. 4651–4664. PMLR, 2021. URL https://proceedings.mlr.press/v139/jaegle21a.html.

Greg Kamradt. Needle in a haystack - pressure testing LLMs. GitHub: https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023. Accessed: 2025.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:106519–106554, 2024. doi: 10.52202/079017-3381.

Patrick Langer et al. OpenTSLM: Time-series language models for reasoning over multivariate medical text- and time-series data. *arXiv Preprints*, 2025. doi: 10.48550/arXiv.2510.02410.

Sirui Li, Shuhan Xiao, Mihir Joshi, Ahmed Metwally, Daniel McDuff, Wei Wang, and Yuzhe Yang. Hearts: Benchmarking llm reasoning on health time series. *arXiv preprint arXiv:2603.06638*, 2026.

Xiaolin Li, Fang Xiang, Rajesh Panicker, Barry Cardiff, and Deepu John. Classification of ECG based on hybrid features using CNNs for wearable applications. *arXiv preprint arXiv:2206.07648*, 2022. doi: 10.48550/arXiv.2206.07648.

Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Christian S Jensen, et al. Tsfm-bench: A comprehensive and unified benchmark of foundation models for time series forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5595–5606, 2025. doi: 10.1145/3711896.3737442.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023. URL https://proceedings.neurips.cc/paper/2023/hash/6dcf277ea32ce3288914faf369fe6de0-Abstract-Conference.html.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Jungwoo Oh, Gyubok Lee, Seongsu Bae, Joon myoung Kwon, and Edward Choi. ECG-QA: A comprehensive question answering dataset combined with electrocardiogram. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=YWJ7Yi4OtH.

OpenAI. gpt-oss-120b gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Simona Petrutiu, Alan V. Sahakian, and Steven Swiryn. Abrupt changes in fibrillatory wave characteristics at the termination of paroxysmal atrial fibrillation in humans. *Europace*, 9(7):466–470, 2007. doi: 10.1093/europace/eum096.

Team Qwen. Qwen3.5-omni technical report, 2026. URL https://arxiv.org/abs/2604.15804.

Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, pp. 1–9, 2008. doi: 10.1109/PHM.2008.4711414.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv Preprints*, 2025. doi: 10.48550/arXiv.2601.03267.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8317–8326, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Singh_Towards_VQA_Models_That_Can_Read_CVPR_2019_paper.html.

José F Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. Deep learning for time series forecasting: a survey. *Big data*, 9(1):3–21, 2021. doi: 10.1109/JSEN.2019.2923982.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun, Zirui Zhuang, Jinming Wu, Lei Zhang, and Jianxin Liao. Chattime: A unified multimodal time series foundation model bridging numerical and textual data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 12694–12702, 2025a. doi: 10.1609/aaai.v39i12.33384.

Hengyi Wang, Haizhou Shi, Shiwei Tan, Weiyi Qin, Wenyuan Wang, Tunyu Zhang, Akshay Nambi, Tanuja Ganu, and Hao Wang. Multimodal needle in a haystack: Benchmarking long-context capability of multimodal large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3221–3241, 2025b. doi: 10.18653/v1/2025.naacl-long.166.

Tao Wang, Changhua Lu, Yining Sun, Mei Yang, Chun Liu, and Chunsheng Ou. Automatic ECG classification using continuous wavelet transform and convolutional neural network. *Entropy*, 23 (1):119, 2021. doi: 10.3390/e23010119.

Weiyun Wang, Shuibo Zhang, Yiming Ren, Yuchen Duan, Tiantong Li, Shuo Liu, Mengkang Hu, Zhe Chen, Kaipeng Zhang, Lewei Lu, et al. Needle in a multimodal haystack. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a. doi: 10.52202/079017-0649.

Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. Videoagent: Long-form video understanding with large language model as agent. In *European Conference on Computer Vision*, pp. 58–76. Springer, 2024b. doi: 10.1007/978-3-031-72989-8_4.

Yilin Wang, Peixuan Lei, Jie Song, Yuzhe Hao, Tao Chen, Yuxuan Zhang, Lei Jia, Yuanxiang Li, and Zhongyu Wei. Itformer: Bridging time series and natural language for multi-modal qa with large-scale multitask dataset. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025c. URL https://icml.cc/virtual/2025/poster/45847.

Matthew Willetts, Sven Hollowell, Louis Aslett, Chris Holmes, and Aiden Doherty. Statistical machine learning of sleep and physical activity phenotypes from sensor data in 96,220 UK Biobank participants. *Scientific Reports*, 8(1):7961, 2018. doi: 10.1038/s41598-018-26174-1.

Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *Proceedings of the VLDB Endowment*, 18(8):2385–2398, April 2025. ISSN 2150-8097. doi: 10.14778/3742728.3742735.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Hugo Yèche, Rita Kuznetsova, Marc Zimmermann, Matthias Hüser, Xinrui Lyu, Martin Faltys, and Gunnar Rätsch. Hirid-icu-benchmark – a comprehensive machine learning benchmark on high-resolution icu data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://arxiv.org/abs/2111.08536.

Yifei Yu, Qian-Wen Zhang, Lingfeng Qiao, Di Yin, Fang Li, Jie Wang, Zengxi Chen, Suncong Zheng, Xiaolong Liang, and Xing Sun. Sequential-niah: A needle-in-a-haystack benchmark for extracting sequential needles from long contexts, 2025. URL https://arxiv.org/abs/2504.04713.

Hang* Yuan, Shing* Chan, Andrew P Creagh, Catherine Tong, Aidan Acquah, David A Clifton, and Aiden Doherty. Self-supervised learning for human activity recognition using 700,000 person-days of wearable data. *NPJ digital medicine*, 7(1):91, 2024. doi: 10.1038/s41746-024-01062-3.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9556–9567, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Yue_MMMU_A_Massive_Multi-discipline_Multimodal_Understanding_and_Reasoning_Benchmark_for_CVPR_2024_paper.html.

Alex L Zhang, Tim Kraska, and Omar Khattab. Recursive language models. *arXiv Preprints*, 2025. doi: 10.48550/arXiv.2512.24601.

Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh K Gupta, and Jingbo Shang. Large language models for time series: A survey. *arXiv Preprints*, 2024. doi: 10.48550/arXiv.2402.01801.

A TS-HAYSTACK DETAILED RESULTS

In the following, we report the per-task accuracy underlying the aggregate plots in Section 4.2, with one table per source dataset (Capture24, Sleep PSG and LTAF). All tables follow the same conventions. All three TSLMs use a Qwen3.5-9B backbone; the ARTS columns name the base LLM that orchestrates the loop. Cells marked "OOM" indicate that the corresponding model exhausted GPU memory at training or inference time for that context length and was not evaluated. Per (task, length) row, the best result across the ten evaluated models is in **bold**; reference baselines are not eligible for bolding. Context length coverage varies by dataset, reflecting the natural durations of each source corpus and the task instantiation choices documented in Appendix B.

ARTS leads on time-stamp retrieval; single-pass models lead on prior-dominated tasks. ARTS performs best at long context lengths on Localization, Counting, Comparison, Multi-hop, and Anomaly Localization across datasets — the tasks where the answer must be specifically retrieved from the sample. On Antecedent and State Query, the picture inverts: Flamingo holds 0.68–0.84 across all four lengths on LTAF State Query, and 0.73–0.78 across Sleep Antecedent, while ARTS sits below in both.

We hypothesize that single-pass models internalize the marginal label distribution and Markovian priors over label transitions, answering correctly without per-segment perception. This is consistent with ARTS, whose inference-only agentic loop does not learn these priors and instead relies on bottom-up tool calls. Additionally, in Capture24, where needle insertion balances the weight of these prior coefficients, ARTS outperforms all TSLMs across context lengths. Nonetheless, further experiments that destabilize these priors and isolate the performance of TSLMs on these tasks remain future work to explain this performance gap between localization-heavy and prior-informed tasks.

Time-interval scoring localizes the bottleneck. The collapse to near-zero on Localization and Multi-hop at longer samples is uniform across all single-pass models. ARTS preserves better accuracy on these tasks by decoupling timeseries classification and reasoning from the sample's context length, but also shows the steepest *relative* degradation, leaving meaningful room for future work on long horizon time-series retrieval.

A.1 PER-DATASET, PER-TASK ACCURACY

We present per-dataset, per-task accuracy across context lengths in Tables 2–4 below.

Table 2: **Capture24-Haystack: per-task accuracy across context lengths.** TSLMs and ARTS orchestrators are evaluated models (best per row in bold); QA-only and Oracle are reference points. "OOM" marks cells the model could not process on a single GPU.

		TSLMs			**ARTS (base LLM)**							**Reference**	
Task	**Length**	ChatTime	ITFormer	Flamingo	Qwen3.5-2B	Qwen3.5-9B	Qwen3.5-35B-A3B	gpt-oss-120B	DeepSeek-V3.2	Gemini-3.8-Flash	GPT-5.4	*QA-only*	*Oracle*
Existence	100 s	OOM	0.53	0.50	0.51	0.25	0.54	**0.59**	0.55	0.52	0.59	0.55	1.00
	15 min	OOM	0.49	0.55	0.21	0.33	0.48	0.57	0.56	0.55	**0.65**	0.54	1.00
	1 h	OOM	0.53	0.50	0.19	0.28	0.35	0.46	0.44	0.49	**0.66**	0.51	1.00
	2 h	OOM	0.54	0.55	0.19	0.43	0.38	0.55	0.56	0.56	**0.67**	0.51	1.00
Localization	100 s	OOM	0.11	0.09	0.09	0.11	0.20	0.14	0.25	0.37	**0.41**	0.00	1.00
	15 min	OOM	0.02	0.00	0.01	0.03	0.04	0.01	0.03	0.06	**0.51**	0.00	1.00
	1 h	OOM	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	**0.51**	0.00	1.00
	2 h	OOM	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.01	**0.55**	0.00	1.00
Counting	100 s	OOM	**0.29**	0.27	0.11	0.03	0.07	0.10	0.11	0.09	0.09	0.19	0.98
	15 min	OOM	0.23	**0.26**	0.04	0.04	0.04	0.05	0.07	0.03	0.22	0.16	0.97
	1 h	OOM	0.19	0.21	0.02	0.02	0.02	0.02	0.01	0.03	**0.31**	0.23	0.95
	2 h	OOM	0.17	0.17	0.01	0.01	0.02	0.03	0.01	0.02	**0.24**	0.21	0.97
Ordering	100 s	OOM	0.46	0.53	0.19	**0.85**	0.56	0.77	0.64	0.57	0.74	0.50	1.00
	15 min	OOM	0.49	0.48	0.63	0.75	0.55	0.62	0.55	0.48	**0.83**	0.43	1.00
	1 h	OOM	0.45	0.41	0.49	0.77	0.63	0.61	0.59	0.51	**0.79**	0.53	1.00
	2 h	OOM	0.50	0.52	0.57	**0.75**	0.59	0.63	0.55	0.53	0.74	0.55	1.00
State Query	100 s	OOM	0.33	0.27	0.13	**0.42**	0.19	0.23	0.35	0.20	0.39	0.31	0.74
	15 min	OOM	0.27	0.31	**0.62**	0.50	0.29	0.30	0.43	0.28	0.51	0.24	0.84
	1 h	OOM	0.29	0.39	0.52	**0.56**	0.37	0.31	0.48	0.27	0.43	0.37	0.84
	2 h	OOM	0.25	0.33	**0.56**	0.55	0.39	0.26	0.45	0.23	0.41	0.25	0.87
Antecedent	100 s	OOM	0.07	0.07	0.17	**0.37**	0.22	0.19	0.17	0.21	0.15	0.09	0.93
	15 min	OOM	0.12	0.13	**0.53**	0.41	0.14	0.07	0.08	0.10	0.27	0.11	0.97
	1 h	OOM	0.09	0.14	**0.52**	0.42	0.17	0.06	0.09	0.07	0.27	0.13	0.90
	2 h	OOM	0.12	0.08	**0.55**	0.37	0.25	0.05	0.08	0.05	0.21	0.12	0.97
Comparison	100 s	OOM	0.18	0.23	0.06	0.20	0.31	0.19	0.29	0.30	**0.41**	0.00	0.95
	15 min	OOM	0.13	0.20	0.03	0.11	0.19	0.11	0.24	0.25	**0.43**	0.00	0.97
	1 h	OOM	0.07	0.17	0.02	0.10	0.16	0.04	0.20	0.21	**0.39**	0.01	0.95
	2 h	OOM	0.15	0.14	0.00	0.07	0.14	0.05	0.21	0.17	**0.35**	0.01	0.95
Multi-hop	100 s	OOM	0.08	0.07	0.05	0.19	0.21	0.09	0.18	0.31	**0.40**	0.00	0.99
	15 min	OOM	0.02	0.01	0.00	0.03	0.07	0.01	0.08	0.01	**0.41**	0.00	0.99
	1 h	OOM	0.00	0.00	0.00	0.01	0.01	0.00	0.01	0.00	**0.31**	0.00	0.99
	2 h	OOM	0.01	0.00	0.00	0.01	0.01	0.01	0.00	0.01	**0.24**	0.00	0.97
Anomaly Det.	100 s	OOM	0.48	0.48	0.19	0.38	0.45	0.49	**0.59**	0.54	0.56	0.50	1.00
	15 min	OOM	0.53	0.53	0.15	0.39	0.38	0.43	0.47	0.50	**0.63**	0.50	1.00
	1 h	OOM	0.53	0.53	0.19	0.32	0.42	0.51	**0.54**	0.49	0.51	0.47	1.00
	2 h	OOM	**0.56**	**0.56**	0.15	0.23	0.39	0.53	0.52	0.43	0.52	0.52	1.00
Anomaly Loc.	100 s	OOM	**0.50**	**0.50**	0.06	0.15	0.29	0.22	0.30	0.37	0.38	0.43	0.99
	15 min	OOM	**0.42**	**0.42**	0.06	0.10	0.19	0.17	0.18	0.28	0.29	0.37	0.99
	1 h	OOM	**0.46**	**0.46**	0.05	0.08	0.19	0.14	0.11	0.23	0.27	0.38	0.99
	2 h	OOM	**0.48**	**0.48**	0.04	0.08	0.13	0.11	0.13	0.27	0.21	0.41	1.00

Table 3: **Sleep-Haystack: per-task accuracy across context lengths.** Anomaly tasks span shorter horizons because arousal events are short and substantially more frequent than sleep-stage epochs (see Appendix C.8). Only instantiated (task, length) pairs are shown. Existence and Anomaly Detection are not instantiated at the longest length and (1 h, 2 h, $\sim$ 8 h) respectively: the target class is near-deterministically present at such lengths.

		TSLMs			**ARTS (base LLM)**							**Reference**	
Task	**Length**	ChatTime	ITFormer	Flamingo	Qwen3.5-2B	Qwen3.5-9B	Qwen3.5-35B-A3B	gpt-oss-120B	DeepSeek-V3.2	Gemini-3.8-Flash	GPT-5.4	*QA-only*	*Oracle*
Existence	15 min	OOM	0.46	0.46	0.77	0.77	0.81	0.88	0.88	0.88	**0.89**	0.51	0.99
	1 h	OOM	0.53	0.53	0.55	0.66	0.66	0.69	0.69	0.71	**0.80**	0.45	1.00
	2 h	OOM	0.46	0.47	0.19	0.55	0.57	0.58	0.58	0.61	**0.63**	0.49	1.00
Localization	15 min	OOM	0.07	0.03	0.15	0.23	0.17	0.17	0.25	0.25	**0.34**	0.07	1.00
	1 h	OOM	0.01	0.01	0.02	0.17	0.14	0.07	0.17	0.17	**0.35**	0.01	1.00
	2 h	OOM	0.02	0.01	0.03	0.08	0.08	0.01	0.09	0.09	**0.13**	0.02	1.00
	9 h	OOM	0.00	0.00	0.01	0.04	0.04	0.01	0.05	**0.06**	**0.06**	0.00	1.00
Counting	15 min	OOM	0.43	0.49	0.42	0.37	0.39	0.45	0.45	0.46	**0.51**	0.43	0.97
	1 h	OOM	0.23	**0.35**	0.05	0.16	0.18	0.18	0.19	0.18	0.33	0.29	0.95
	2 h	OOM	**0.19**	0.19	0.02	0.09	0.08	0.07	0.11	0.09	0.10	0.19	0.95
	9 h	OOM	**0.05**	0.02	0.00	0.01	0.03	0.03	0.04	0.04	0.03	0.03	0.97
Ordering	15 min	OOM	0.59	0.68	0.71	0.72	0.73	0.72	0.73	0.72	**0.77**	0.63	0.97
	1 h	OOM	0.66	0.64	0.23	0.73	0.71	0.72	0.75	0.74	**0.81**	0.72	1.00
	2 h	OOM	0.65	0.64	0.19	**0.74**	0.70	0.71	**0.74**	0.71	0.71	0.72	1.00
	9 h	OOM	0.61	0.55	0.22	0.65	0.62	0.68	0.66	**0.69**	0.67	0.75	0.99
State Query	15 min	OOM	0.48	0.51	0.34	0.48	0.49	0.37	0.46	0.50	**0.58**	0.52	0.84
	1 h	OOM	0.41	0.43	**0.73**	0.43	0.42	0.29	0.37	0.43	0.62	0.43	0.95
	2 h	OOM	0.49	0.48	**0.73**	0.35	0.34	0.27	0.27	0.31	0.37	0.48	0.94
	9 h	OOM	0.44	0.50	**0.69**	0.46	0.35	0.25	0.30	0.33	0.36	0.50	0.92
Antecedent	15 min	OOM	**0.79**	0.78	0.41	0.58	0.49	0.47	0.43	0.53	0.60	0.77	0.89
	1 h	OOM	0.72	0.73	**0.75**	0.41	0.41	0.37	0.37	0.45	0.53	0.71	0.91
	2 h	OOM	0.72	0.73	**0.75**	0.53	0.58	0.53	0.51	0.61	0.49	0.73	0.89
	9 h	OOM	0.69	**0.73**	0.69	0.47	0.49	0.40	0.34	0.49	0.45	0.75	0.89
Comparison	15 min	OOM	0.03	0.10	0.18	0.25	0.29	0.14	0.32	0.32	**0.39**	0.07	0.85
	1 h	OOM	0.01	0.01	0.04	0.28	0.29	0.13	0.30	0.32	**0.40**	0.00	0.89
	2 h	OOM	0.00	0.00	0.07	0.25	0.26	0.11	0.29	0.30	**0.36**	0.00	0.91
	9 h	OOM	0.01	0.00	0.01	0.24	0.26	0.09	0.28	0.28	**0.30**	0.00	0.98
Multi-hop	15 min	OOM	0.05	0.02	0.11	0.21	0.17	0.11	0.19	0.21	**0.26**	0.05	0.82
	1 h	OOM	0.01	0.01	0.03	0.15	0.14	0.05	0.15	**0.18**	0.17	0.01	0.73
	2 h	OOM	0.01	0.01	0.01	0.07	0.06	0.03	0.06	0.07	**0.07**	0.01	0.70
	9 h	OOM	0.01	0.00	0.00	0.02	**0.03**	0.01	**0.03**	0.03	0.02	0.00	0.68
Anomaly Det.	100 s	OOM	0.50	0.49	0.19	0.65	0.63	0.63	0.67	0.67	**0.69**	0.48	1.00
	15 min	OOM	0.51	0.47	0.21	0.60	0.61	0.61	0.61	0.61	**0.63**	0.51	1.00
Anomaly Loc.	100 s	OOM	0.33	0.38	0.10	0.32	0.27	0.25	0.37	0.34	**0.41**	0.41	1.00
	15 min	OOM	0.03	0.03	0.03	0.05	0.04	0.02	0.05	0.07	**0.09**	0.02	1.00
	1 h	OOM	0.00	0.02	0.01	0.05	0.05	0.01	0.05	0.07	**0.09**	0.01	1.00
	9 h	OOM	0.00	0.00	0.00	0.01	0.01	0.00	0.01	0.01	**0.01**	0.00	0.96

Table 4: **LTAF-Haystack: per-task accuracy across context lengths.** Only instantiated (task, length) pairs are shown. Existence and Anomaly Detection are not instantiated at 2 h and (1 h, 2 h) respectively: the target classes are near-deterministically present at these horizons.

		TSLMs			**ARTS (base LLM)**							**Reference**	
Task	**Length**	ChatTime	ITFormer	Flamingo	Qwen3.5-2B	Qwen3.5-9B	Qwen3.5-35B-A3B	gpt-oss-120B	DeepSeek-V3.2	Gemini-3.8-Flash	GPT-5.4	*QA-only*	*Oracle*
Existence	100 s	OOM	0.46	0.52	0.65	0.69	0.65	0.81	0.78	0.82	**0.83**	0.53	1.00
	15 min	OOM	0.53	0.51	0.21	0.71	0.69	0.79	0.77	**0.87**	0.87	0.53	1.00
	1 h	OOM	0.57	0.44	0.17	0.71	0.64	0.74	0.78	0.80	**0.84**	0.46	1.00
Localization	100 s	OOM	0.04	0.05	0.14	0.20	0.21	0.30	0.32	0.30	**0.32**	0.02	0.99
	15 min	OOM	0.01	0.00	0.04	0.22	0.18	0.24	0.25	**0.31**	0.29	0.00	0.99
	1 h	OOM	0.00	0.00	0.01	0.13	0.19	0.15	0.23	**0.28**	**0.28**	0.00	0.99
	2 h	OOM	0.00	0.00	0.01	0.12	0.13	0.13	0.19	**0.26**	**0.26**	0.00	0.99
Counting	100 s	OOM	0.29	0.34	0.32	0.37	0.41	0.49	0.48	**0.52**	0.49	0.34	0.99
	15 min	OOM	0.27	0.29	0.07	0.32	0.31	0.35	0.38	0.41	**0.41**	0.31	0.95
	1 h	OOM	0.25	0.27	0.05	0.25	0.28	0.31	0.33	0.34	**0.47**	0.27	1.00
	2 h	OOM	0.24	0.29	0.05	0.21	0.28	0.28	0.29	0.32	**0.41**	0.30	0.97
Ordering	100 s	OOM	**0.76**	0.73	0.60	0.63	0.55	0.67	0.68	0.65	0.65	0.71	1.00
	15 min	OOM	**0.78**	0.77	0.24	0.57	0.57	0.60	0.61	0.62	0.68	0.83	0.99
	1 h	OOM	**0.75**	0.68	0.16	0.55	0.55	0.65	0.62	0.65	0.63	0.77	1.00
	2 h	OOM	**0.75**	0.74	0.17	0.53	0.49	0.56	0.57	0.62	0.59	0.79	1.00
State Query	100 s	OOM	0.67	0.68	0.49	0.93	0.84	**0.95**	0.93	0.94	0.93	0.65	1.00
	15 min	OOM	0.77	0.76	0.79	0.81	0.79	0.83	**0.87**	0.86	0.84	0.77	1.00
	1 h	OOM	0.79	0.78	0.76	0.85	0.79	**0.87**	**0.87**	0.86	0.85	0.80	1.00
	2 h	OOM	0.84	0.84	0.79	0.83	**0.88**	0.83	0.85	**0.88**	0.85	0.83	0.98
Antecedent	100 s	OOM	0.46	0.50	0.47	**0.68**	0.62	0.63	0.61	0.48	0.59	0.53	1.00
	15 min	OOM	0.45	0.47	**0.81**	0.63	0.61	0.59	0.65	0.57	0.57	0.49	1.00
	1 h	OOM	0.48	0.49	**0.79**	0.56	0.58	0.58	0.57	0.55	0.52	0.52	0.99
	2 h	OOM	0.51	0.57	**0.76**	0.64	0.65	0.59	0.57	0.61	0.58	0.53	0.99
Comparison	100 s	OOM	0.06	0.01	0.18	0.34	0.37	0.42	0.43	0.45	**0.47**	0.10	0.90
	15 min	OOM	0.02	0.06	0.04	0.31	0.27	0.37	0.36	0.39	**0.40**	0.03	0.89
	1 h	OOM	0.02	0.07	0.02	0.19	0.19	0.23	0.28	0.30	**0.33**	0.00	0.89
	2 h	OOM	0.03	0.17	0.01	0.19	0.25	0.28	0.29	**0.35**	0.31	0.00	0.91
Multi-hop	100 s	OOM	0.16	0.13	0.07	0.15	0.17	0.23	0.15	**0.30**	0.28	0.15	0.87
	15 min	OOM	0.01	0.03	0.03	0.15	0.08	0.14	0.17	**0.23**	0.19	0.00	0.84
	1 h	OOM	0.00	0.09	0.01	0.13	0.13	0.13	0.12	0.21	**0.21**	0.00	0.83
	2 h	OOM	0.00	0.01	0.00	0.07	0.09	0.09	0.11	0.13	**0.17**	0.00	0.83
Anomaly Det.	100 s	OOM	0.75	0.75	0.24	0.70	0.71	0.71	0.70	**0.77**	0.72	0.75	1.00
	15 min	OOM	0.58	0.57	0.19	0.56	0.56	0.57	**0.59**	**0.59**	0.59	0.58	1.00
Anomaly Loc.	100 s	OOM	0.02	0.01	0.17	0.73	0.59	0.75	0.75	0.75	**0.76**	0.01	1.00
	15 min	OOM	0.00	0.00	0.09	0.49	0.47	0.51	**0.51**	**0.51**	0.51	0.00	1.00
	1 h	OOM	0.00	0.00	0.02	0.14	0.16	0.17	0.17	0.17	**0.19**	0.00	0.91
	2 h	OOM	0.00	0.00	0.01	0.10	0.09	0.09	0.11	0.11	**0.12**	0.00	0.87

B APPENDIX: TS-HAYSTACK

In the following, we present TS-Haystack's construction and report per-dataset statistics. We first give a high-level overview of the benchmark (Appendix B.1), then describe shared components of the generation pipeline: the needle-insertion protocol and its statistical validation (Appendix B.2), task templates and ground-truth derivation (Appendix C.12), evaluation metrics, and random-baseline calculation. Each of the three source datasets (Capture24, Sleep PSG, LTAF) is then detailed in its own subsection, covering the source corpus, the haystack generation pipeline used to repurpose it, illustrative samples, and the corresponding ARTS classifier tool.

B.1 TS-HAYSTACK OVERVIEW

TS-Haystack is built from three openly available, expert-annotated physiological time-series corpora that span three distinct modalities: wrist accelerometry, multi-channel polysomnography, and ECG. Each source dataset is repurposed into a subset of TS-Haystack's ten retrieval and understanding tasks, instantiated at a discrete grid of context lengths. In aggregate, the benchmark is composed of more than 12k hours of source recordings augmented to controlled retrieval samples with contexts spanning multiple orders of magnitude (100 s up to $\sim$ 9 h).

Table 5 summarizes the source corpora and the context-length grid instantiated per dataset. Table 6 reports which of the ten tasks are instantiated at which context lengths per dataset, together with the total number of generated (train / validation / test) samples. Two systematic exclusions are worth noting up front:

- *Existence* and *Anomaly Detection* are not instantiated at the longest context length for some datasets, because the queried target class is near-deterministically present in any naturally drawn segment at that horizon, collapsing the answer to a trivial "Yes".
- On Sleep PSG, anomaly tasks (Anomaly Detection / Localization) operate on the *arousal* annotation track rather than the sleep-stage track, and are therefore instantiated at shorter horizons (down to 100 s) to account for the higher density and shorter duration of arousal events (Appendix C.8).

Table 5: **TS-Haystack source datasets.** Per-dataset modality, channel count, sampling rate, number of source recordings, source corpus duration, total generated benchmark duration (summed over tasks and context lengths), and the context-length grid at which the source is instantiated in TS-Haystack. *Construction* indicates whether haystack samples are produced via semi-synthetic needle insertion or natural-segment sampling (Section 3.2).

Source dataset	Modality	Ch.	f_s (Hz)	# Rec.	Source hrs	Benchmark hrs	Context lengths	Construction
Capture24 (Chan et al., 2024)	Wrist accelerometer	3	100	151	$\sim$3,883	42,611	100 s, 15 min, 1 h, 2 h	Semi-synthetic
Sleep PSG (Ghassemi et al., 2018)	Polysomnography	13	100	994	$\sim$6,958	103,963	100s, 15 min, 1 h, 2 h, 9 h	Natural
LTAF (Petrutiu et al., 2007)	ECG	2	128	84	$\sim$1,961	36,108	100 s, 15 min, 1 h, 2 h	Natural
Total				1,229	$\sim$12,802	182,682		

Table 6: **TS-Haystack task instantiation matrix.** For each (dataset, context length) cell we report which of the ten tasks are instantiated (✓) or excluded (—), together with the number of generated (train / validation / test) samples per (task, length) cell. Task abbreviations: **Ex** existence, **Loc** localization, **Cnt** counting, **Ord** ordering, **SQ** state query, **Ant** antecedent, **Cmp** comparison, **MH** multi-hop, **AD** anomaly detection, **AL** anomaly localization. Per-cell sample counts apply to each instantiated task in that row.

		Direct retrieval			**Temporal retrieval**			**Multi-step**		**Anomaly**		
Dataset	**Length**	Ex	Loc	Cnt	Ord	SQ	Ant	Cmp	MH	AD	AL	**# samples (tr/va/te)**
Capture24	100 s	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
	15 min	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
	1 h	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
	2 h	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
Sleep PSG	100 s	—	—	—	—	—	—	—	—	✓	✓	1000 / 150 / 150
	15 min	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
	1 h	✓	✓	✓	✓	✓	✓	✓	✓	—	✓	1000 / 150 / 150
	2 h	✓	✓	✓	✓	✓	✓	✓	✓	—	—	1000 / 150 / 150
	9 h	—	✓	✓	✓	✓	✓	✓	✓	—	✓	1000 / 150 / 150
LTAF	100 s	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
	15 min	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	1000 / 150 / 150
	1 h	✓	✓	✓	✓	✓	✓	✓	✓	—	✓	1000 / 150 / 150
	2 h	—	✓	✓	✓	✓	✓	✓	✓	—	✓	1000 / 150 / 150

B.2 Needle Insertion

Needle insertion is the construction protocol that lets TS-Haystack turn a single annotated corpus into a controlled, semi-synthetic, grid of (context length, target class, needle count) cells, by embedding segments of a target signal (*needles*) into longer recordings of the same domain (*haystacks*). The protocol decouples the *what* happens (the needle) to then *when* and *within how much surrounding context* it happens (the haystack). This decoupling is what enables arbitrary context-length scaling, explicit difficulty control, and combinatorial sample diversity. One of the three source datasets in TS-Haystack admits this protocol (Capture24); the other two (Sleep PSG, LTAF) do not, and instead rely on natural-segment sampling. Whether a domain admits the protocol turns on a single property of the source signal that we call *local context-independence*.

Operationalization of context-independence. Let $\mathbf{x}_{[s:e]}$ denote a bout of class c within a recording $\mathbf{X}$, and let $\mathbf{X}_{\setminus[s:e]}$ denote the rest of the recording. We say the source signal is *locally context-independent* for c if the bout's distribution conditional on its surrounding context is well-approximated by its marginal distribution given the class label,

$$p\big(\mathbf{x}_{[s:e]} \mid \mathbf{X}_{\setminus[s:e]},\, c\big) \;\approx\; p\big(\mathbf{x}_{[s:e]} \mid c\big). \tag{1}$$

When (1) holds, transplanting a bout into a different background of the same domain preserves its semantics and yields a composite whose joint distribution is statistically close to that of naturally observed class-c bouts embedded in arbitrary contexts. Activity bouts on the wrist satisfy this condition to a usable degree of approximation, with the residual gap quantifiable by a domain-specific validation check. This assumption may not hold for all time-series domains. In the case of sleep stages, these signals are embedded in the cyclic NREM/REM macrostructure scored by AASM rules (Berry et al., 2017). In the case of ECG recordings, the interpretation of an arrhythmia depends on surrounding inter-beat morphology and clinical guidelines requiring multiple seconds of contiguous rhythm (Hindricks et al., 2021) surrounding the labeled section. Thus, insertion would produce an out-of-distribution composite that no boundary-smoothing operation can repair, and we fall back to natural-segment sampling. Further details for needle insertion to validate this hypothesis can be found in the dataset specific sections below.

C EXPERIMENTAL SETUP

C.1 TSLM TRAINING DETAILS

We aim to train the single-pass TSLMs (Flamingo, ITFormer and ChatTime) with a single, dataset-agnostic recipe so that any cross-dataset or cross-architecture difference reflects representational capacity rather than tuning. Our experiments are conducted on a single NVIDIA GH200 GPU (96 GB), with the out-of-memory policy described below. Per-architecture hyperparameters are summarized in Table 7.

Optimizer, schedule, and trainable surface. AdamW (Loshchilov & Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 0.01, learning rate 2×10^{-4} for every trainable parameter group, and gradient clipping at L2-norm 1.0. The schedule is a linear warmup over the first 3% of total steps followed by linear decay to zero (Hugging Face `get_linear_schedule_with_warmup`). We train for up to 5 epochs with early stopping on validation loss (patience $= 3$) and retain the best-validation checkpoint for test set evaluation. Evaluation generates up to 500 new tokens per sample and runs on the full validation and test splits. The Qwen3.5-9B (Qwen, 2026) backbone is frozen in every configuration except ChatTime, as per the original implementation.

Per-architecture variants.

- **Flamingo** (Langer et al., 2025): a Perceiver-Resampler projector with 64 latents feeds gated cross-attention layers inserted into the backbone at every layer ($\text{cross_attn_every_n_layers} = 1$). Trainable: the resampler, the cross-attention adapters, and the LM input embeddings.
- **ITFormer** (Wang et al., 2025c): a 2-layer instruction-aware time-series transformer ($d_{\text{model}} = 2048$, 16 heads, dropout 0.1) followed by three small projections (`ts_project`, `query_project`, `fusion_project`) and a 25-token learnable prefix prepended to the LM.
- **ChatTime** (Wang et al., 2025a): a pure-LLM TSLM that discretises each time-series value into one of $10\,002$ bin tokens appended to the LM tokenizer. Because the bin vocabulary is added to the tokenizer, ChatTime requires a structurally different training mode: the new token embeddings must be learnable, and the LM itself receives LoRA adapters ($r = 16$, $\alpha = 16$, dropout $= 0$, applied to all attention and MLP projections).

Time-series encoders. Every architecture keeps its shipped encoder. Flamingo uses OpenTSLM's convolutional patch tokenizer (patch size 4, embedding dimension 128), trained jointly with the resampler; its learned positional embeddings are extended to the number of patches of the longest window in each dataset. ITFormer uses the patch-transformer encoder released with ITFormer-3B (patch length 60), kept frozen and applied per 600-sample chunk as in the original implementation, with the resulting patch tokens concatenated before the fusion module. ChatTime retains its value-binning tokenizer. No encoder truncates or decimates its input; a (task, length) cell whose windows do not fit in GPU memory is reported as OOM.

Base prompt template. Each sample uses a shared three-part skeleton, populated by dataset-specific question/answer-format fields. The *pre-prompt* names the modality, sampling rate, recording duration, and absolute time range so that all timestamps in the question and answer are interpretable relative to a known clock. The *question* is drawn from the QA shard. The *post-prompt* embeds an answer-type-specific format hint (boolean / integer / category / time_range) and instructs the model to terminate its response with the literal string `"Answer: <your answer>"` so the evaluator can extract the prediction by regex. The time-series payload is inserted between pre- and post-prompt as one `TextTimeSeriesPrompt(<channel description>, <numerical signal>)` per channel (e.g. "Triaxial accelerometer (g), x/y/z" for Capture24).

Prompt example. To make the skeleton concrete, consider a Capture24-Haystack `existence` task at the 1-hour context length. The assembled prompt is:

```
You are given accelerometer data in all three dimensions
from a wrist-worn sensor.  The recording spans from
7:05:59.007 AM to 8:05:59.007 AM.
Question:  Check whether this recording contains vehicle.
⟨TextTimeSeries:  "Triaxial accelerometer (g), x", x∈ℝ^360000⟩
⟨TextTimeSeries:  "Triaxial accelerometer (g), y", x∈ℝ^360000⟩
⟨TextTimeSeries:  "Triaxial accelerometer (g), z", x∈ℝ^360000⟩
Instructions:
- Analyze the accelerometer data carefully.
- Think step-by-step about what the signal patterns
indicate.
- Write your reasoning as a natural paragraph.
- End your response with "Answer:  ⟨your answer⟩"
```

Context-length packing and OOM handling. The benchmark spans context lengths varying by two to three orders of magnitude per dataset. Batches are grouped by context length, so a window is never zero-padded to the length of a longer one, and every architecture trains with `batch_size` $= 1$; a single window is therefore the smallest unit that can exhaust GPU memory. A batch that raises a CUDA out-of-memory error is recorded and skipped, and training continues on the remaining windows.

Reproducibility. Every config sets `seed: 42` and dataset constructors derive deterministic per-(task, context, split, sample-index) seeds, so the train/val/test parquet shards are bit-identical across re-runs; the only run-to-run variance is the training stochasticity of the trainable bridging modules. Deterministic RNG is set in PyTorch, NumPy, and CUDA.

Table 7: Per-architecture training hyperparameters. All runs share $\mathrm{lr} = 2 \times 10^{-4}$, AdamW $\mathrm{wd} = 0.01$, $\|g\|_2 \leq 1$, linear warmup ratio 0.03, max 5 epochs with early stopping (patience 3) on validation loss, seed 42. Backbone is Qwen3.5-9B.

	Flamingo	ITFormer	ChatTime
TS encoder	CNN tokenizer (trained)	ITFormer-3B patch enc. (frozen)	— (text)
Bridging module	Perceiver-Resampler	25-token prefix	LoRA on LM
LM backbone	frozen	frozen	LoRA (r=16, α=16)
LM input embeddings	trained	trained	trained
Default batch size	1	1	1

C.2 ARTS DETAILS

Pre-pass rendering. The classifier pre-pass sweeps g_ϕ over a non-overlapping tiling of the recording at a fixed window size and returns per-tile predictions. Before reaching the orchestrator, this output is post-processed and rendered into text. Adjacent tiles sharing a label are compacted into bouts. The bout list is then rendered as a Markdown table (one line per bout, with per-bout peak-amplitude or mean-confidence aggregates depending on the domain) and inserted between the standard pre-prompt (modality, sampling rate, duration, time range) and the question. The orchestrator never receives a raw waveform or sample-level vector, only this rendered timeline plus the question. The total textual size of the pre-pass scales with the number of detected events rather than the recording length, which is what makes the agentic loop tractable on multi-hour windows.

Prompt example. We showcase the prompt example at the same 1-hour context length used for the TSLM prompt example above. The full prompt assembled for the orchestrator is:

```
System:
You are given accelerometer data in all three dimensions
from a wrist-worn sensor.

User (pre):
You are given accelerometer data in all three dimensions
```

```
from a wrist-worn sensor.  The recording spans from
7:05:59.007 AM to 8:05:59.007 AM.

Predicted Activity Timeline (HAR classifier, 10 s stride):
Recording:  7:05:59.007 AM to 8:05:59.007 AM
7:05:59.007 AM - 7:06:49.007 AM: sitting (conf 0.82)
7:06:49.007 AM - 7:06:59.007 AM: sleep (conf 0.87)
7:06:59.007 AM - 7:07:29.007 AM: sitting (conf 0.60)
... (137 bouts in total)
7:14:39.007 AM - 7:14:49.007 AM: vehicle (conf 0.99)
7:14:49.007 AM - 7:14:59.007 AM: household-chores (conf
0.33)
7:15:09.007 AM - 7:15:59.007 AM: bicycling (conf 0.95)
...
7:24:59.007 AM - 7:27:29.007 AM: vehicle (conf 0.99)
...
Question:  Check whether this recording contains vehicle.
```

User (post):

```
Instructions:
- A classifier-predicted activity timeline (HAR classifier,
10 s stride; predictions MAY BE WRONG) is provided above.
Use it as your starting point and refine any region with the
classify_bout tool when the timeline is uncertain or coarse.
- Use the classify_bout tool to classify time bouts by
providing start and end timestamps.  Bouts longer than 10
seconds are split into non-overlapping 10-second chunks,
each classified independently with the top predicted
activity and its confidence.
- The classifier may make mistakes -- cross-check results
and query more bouts if needed.
- For boolean questions answer 'Yes' or 'No'.  For counting
questions answer with just the number.  For time-based
answers provide a range:  "H:MM:SS.mmm AM/PM to H:MM:SS.mmm
AM/PM".
- Valid activity classes:  bicycling, household-chores,
manual-work, mixed-activity, sitting, sleep, sports,
standing, vehicle, walking.
- This is a long recording (60 minutes).  Max query length
is 120 s.  Use a divide-and-conquer strategy.
- When ready, provide your final answer prefixed with
"Answer:  ".
```

Orchestrators. Table 8 lists the base LLMs used as orchestrators. All share the same classifier tools, pre-pass, prompts and tool budget (15 rounds); the only variable is the LLM. Open-weight models are served through an OpenAI-compatible tool-calling API (Prime Inference); GPT-5.4 through the OpenAI API. Every model is used zero-shot with the provider's default sampling temperature and reasoning settings, so that no orchestrator receives model-specific tuning. Each orchestrator is evaluated on the full 150 test rows of every (task, length) cell.

Table 8: ARTS orchestrators. Parameter counts as published by the model providers (total / active for mixture-of-experts models; "n/a" where undisclosed).

Orchestrator	Identifier	Weights	Parameters
Qwen3.5-2B	`qwen/qwen3.5-2b`	open	2B
Qwen3.5-9B	`qwen/qwen3.5-9b`	open	9B
Qwen3.5-35B-A3B	`qwen/qwen3.5-35b-a3b`	open	35B / 3B
gpt-oss-120b	`openai/gpt-oss-120b`	open	117B / 5.1B
DeepSeek-V3.2	`deepseek/deepseek-v3.2`	open	671B / 37B
Gemini 3.8 Flash	`google/gemini-3.8-flash`	proprietary	n/a
GPT-5.4	`gpt-5.4-2026-03-05`	proprietary	n/a

C.3 Construction of the degradation figure

Figure 5 summarizes accuracy across separately generated benchmark cells at different recording lengths. The left panel plots each task against *its own shortest instantiated context and relative to this length's accuracy*. After this re-centring, every task begins at zero and only the change with context remains. The horizontal axis is likewise relative, so a task at 10 is being asked over a recording ten times longer than its own shortest, which may differ from absolute context length when a given task is not instantiated at 100 s (see Table 6 in Appendix B).

The right panel fits one slope per task and per model, then reports each model's mean slope with a 95% interval obtained by resampling tasks with replacement. Across all nine models, 221 of the 270 task slopes are negative, and the mean slope over tasks is -6.6 points per decade with a 95% interval of $[-9.5, -4.0]$.

C.4 Reference baselines

The results tables report two reference points per cell. The *oracle* uses GPT-5.4 to answer each question from the ground-truth annotations rendered as a timestamped event list, without access to the signal. The *signal-blind* baseline is a Qwen3.5-9B model fine-tuned on question–answer pairs from the benchmark training split, with the signal withheld.

Scoring. Discrete-answer tasks are scored by exact match after type-aware normalization. Time range tasks are scored as correct if $\mathrm{IoU}(\hat{y}, y) \geq 0.25$. For `anomaly_localization`, if both prediction and ground truth begin with "no", the sample is marked correct without parsing intervals.

C.5 Capture24-Haystack

Capture24: base dataset. We use Capture24 (Chan et al., 2024) in its public release: 151 participants each wearing an Axivity AX3 wrist accelerometer at 100 Hz for approximately 24 hours of free-living recording, totalling $\sim$3,883 source-hours. Each sample is a 3-channel acceleration vector in units of g. Bouts are annotated against the Willetts et al. (2018) ten-class scheme: *sleep*, *sitting*, *standing*, *walking*, *bicycling*, *vehicle*, *household-chores*, *manual-work*, *sports*, and *mixed-activity*. The class distribution is heavily skewed: *sleep* and *sitting* together account for over 70% of naturally occurring time, while *bicycling*, *sports*, and *manual-work* each contribute under 2%. We retain the original participant-level partition into train / validation / test splits and draw all needles and backgrounds for each TS-Haystack split from the corresponding Capture24 split, so that no participant contributes to both a haystack background and any needle in a different split.

Capture24-Haystack: benchmark statistics. Capture24-Haystack instantiates all ten TS-Haystack tasks at four context lengths: 100 s, 15 min, 1 h, 2 h, with a fixed $1{,}000 \,/\, 150 \,/\, 150$ train / validation / test sample budget per (task, length) cell (Table 6). The construction draws on 151 participants and spans $42{,}611$ aggregate sample-hours across the benchmark. Aggregate per-task accuracy is reported in Table 2, with a fuller per-domain discussion in Appendix A.

Construction pipeline. We construct each Capture24-Haystack sample as a deterministic function of the (task, context length, sample index, split) tuple via a seeded random-number generator, ensuring bit-identical reproducibility across re-runs. The shared three-step procedure is:

1. **Background selection.** A target context length L defines the haystack window. We sample a participant from the appropriate split, draw a contiguous L-second window from a recording in which the queried target class c is *absent* for the entire window, and adopt this window as the haystack background.
2. **Needle sampling.** For each of n needle slots required by the task (Appendix C.12), we sample an annotated bout of class c from a different participant. Bouts are drawn uniformly at random from the pool of valid class-c bouts that exceed a minimum duration; the sampled bout is then trimmed to a *fixed* per-(task, length) needle width $w_{c,L}$ shared across all samples in that cell.

3. **Insertion.** Each needle is inserted at a controlled non-overlapping location within $[0, L]$ that determines the ground-truth answer. This needle replaces the corresponding background segment after transformations (see Appendix B.2.

Fixed needle width. Needle widths $w_{c,L}$ are either naturally bounded by their full length (Sleep, ECG) or fixed in the generator (Capture24) rather than scaled with L. Equivalently, the needle-to-background ratio $w_{c,L}/L$ *decreases* with L, which mirrors the natural sparsity of relevant events in long recordings.

C.6 CAPTURE24: INSERTION QUALITY VALIDATION

We test a specific property of insertion quality: whether a same-activity splice introduces a detectable local seam. The same mean-shift normalization and cosine-blending operation is used when inserting bouts into longer benchmark recordings, so a short-window test directly probes that local operation. It does not test whether entire composite recordings follow the distribution of natural recordings, as semi-synthetic insertion intentionally disrupts natural transition priors.

Methodology. We construct a binary classification task:

- **Class 0 (negative):** Pure background windows with no insertion.
- **Class 1 (positive):** Pure background windows with a same-activity needle inserted from a *different participant*.

Using same-activity needles isolates insertion artifacts from activity-class differences. If the classifier succeeds, it should detect blending imperfections rather than activity signatures.

We generate 5,000 training samples and 500 test samples at 2 and 3 seconds context length (200 and 300 samples at 100Hz), with needle lengths ranging from 2–8% of context aligned with the true dataset configurations and balanced class distributions. An XGBoost classifier (Chen & Guestrin, 2016) is trained on flattened triaxial accelerometer features (600 and 900 dimensions) using 100 estimators with max depth 6.

Results. The classifier achieves AUCs of 0.499 and 0.490 for the two local window lengths, close to chance. Thus, this XGBoost probe finds no detectable local seam artifact from our same-activity insertion procedure. We therefore find no evidence that this XGBoost probe detects the local same-activity splice under the tested settings. Because the full-length generator uses the same local insertion operation, this result is relevant to seams in longer benchmark recordings. These results point to seam quality under the tested settings.

Limitations. An XGBoost classifier on handcrafted features is a finite-capacity probe; a more expressive classifier might achieve $\mathrm{AUC} > 0.5$ on Capture24 where the present test does not. Moreover, matching activity labels tests local splice detectability, not the contextual plausibility of cross-activity insertions. The latter intentionally alter transition patterns and may differ from natural recordings.

C.7 ARTS CLASSIFIER CONSTRUCTION (CAPTURE24)

The ARTS classifier tool g_ϕ for Capture24 is a frozen-encoder with a trainable classification head, finetuned once on the Capture24 train split with the Willetts et al. (2018) ten-class label scheme (Appendix C.5) and reused unchanged across all Capture24-Haystack tasks. We fix the deployed inference window at 10 s at 30 Hz (the ARTS pre-pass tiles the recording at this resolution). To make the head robust to short bouts that would otherwise be discarded at this tile size, training uses a mixture of window sizes: each batch contains windows of 1 s, 3 s, 6 s, and 10 s, all zero-padded to the encoder's 300-sample (10 s @ 30 Hz) input grid, so the encoder also learns representations on short, padded inputs. Final test evaluation uses 10 s windows only, matching deployment, yet robust variable length classification remains important for the agentic tool use performance.

We compare three frozen-encoder choices for g_ϕ to motivate the final architecture used in ARTS. *Chronos-2* (Ansari et al., 2025) applied here as a frozen encoder feeding a 2-layer MLP head.

OxWearables (Yuan et al., 2024) is a domain-specific wrist-accelerometer encoder pretrained on UK Biobank self-supervised contrastive objectives, applied identically with a 2-layer MLP head on its output. *Dual* concatenates the frozen Chronos-2 and OxWearables embeddings and feeds the joint representation to a 2-layer MLP head. All three configurations share the same head architecture, optimizer (AdamW with weight decay 10^{-4}), learning rate (10^{-3}), batch size (128), class-balanced cross-entropy loss with weights inversely proportional to training class frequencies, cosine learning-rate schedule, and 50-epoch budget with best-validation checkpoint selection on macro-F1. The encoders remain frozen throughout; only the head is trained.

Table 9 reports the resulting test accuracy at the 10 s deployment window and at 30 s for reference. The dual encoder consistently improves classification. We adopt the dual configuration for all Capture24-Haystack ARTS evaluations reported in the main paper.

Table 9: ARTS classifier ablation on Capture24. Three frozen-encoder choices (Chronos-2, OxWearables, dual concatenation) with an identical 2-layer MLP head, evaluated at the 10 s deployed inference window and the 30 s window for reference. Best per (window, metric) cell in **bold**. Macro-F1 is reported alongside plain accuracy because of Capture24's heavy class imbalance.

Window	Encoder	Macro-F1	Accuracy
10 s (deployed)	Chronos-2	0.414	0.665
	OxWearables	0.481	0.690
	Dual	**0.485**	**0.713**
30 s	Chronos-2	0.417	0.663
	OxWearables	0.481	0.690
	Dual	**0.483**	**0.712**

C.8 SLEEP-HAYSTACK

Sleep PSG: base dataset. We use the PhysioNet/Computing in Cardiology Challenge 2018 "You Snooze, You Win" release (Ghassemi et al., 2018): 994 subjects each recorded with a 13-channel polysomnogram at 200 Hz for one in-lab overnight session. Recording lengths range from 5.1 h to 9.5 h (mean 7.71 h, median 7.74 h), totalling $\sim$6,958 source-hours once non-scoring tail segments are excluded. The signal stack consists of six EEG derivations (F3-M2, F4-M1, C3-M2, C4-M1, O1-M2, O2-M1), one EOG channel (E1-M2), one submental EMG (Chin1-Chin2), two inductive respiratory belts (ABD, CHEST), an oronasal AIRFLOW signal, peripheral SaO_2, and a single ECG lead. We decimate to 100 Hz once in the loader to match the sampling regime of the rest of TS-Haystack.

The dataset ships with two parallel annotation regimes that we treat as two distinct label families:

- **Sleep stages**: AASM 5-class scoring (Berry et al., 2017) in 30 s epochs: *Wake*, *N1*, *N2*, *N3*, *REM*. Pooled time fractions are 18.1 / 14.6 / 43.1 / 11.5 / 12.7% respectively, with $N2$ dominating and $N3$/REM sparser but architecturally structured. 90.8% of subjects exhibit all five stages.
- **Arousals**: sample-level event annotations for respiratory and EEG arousals: *rera*, *hypopnea*, *obstructive_apnea*, *central_apnea*, *mixed_apnea*. Median total arousals per subject is 150.5, with median inter-event gap 25.6 s (92.4% of gaps under 5 min).

We split subjects deterministically (seed 42) at the participant level into 695 / 149 / 150 train / validation / test, shared across both label families and all tasks. No subject contributes to more than one split.

Sleep-Haystack: benchmark statistics. Sleep-Haystack uses sleep-stage annotations as its primary label family and arousal annotations only where the underlying clinical structure demands it. All ten TS-Haystack tasks are instantiated on the sleep-stage timeline at four context lengths: 15 min, 1 h, 2 h, and 9 h (whole-night), with three exceptions that route through arousals:

- `anomaly_detection` and `anomaly_localization`, which target arousals because in PSG "anomaly" clinically refers to respiratory and EEG arousals (*rera*, *hypopnea*, *obstructive_apnea*, *central_apnea*, *mixed_apnea*) rather than to a stage. These two tasks operate

on shorter horizons where arousal density is sufficient for retrieval without the target class saturating the window.

- `state_query` ("what sleep stage at the time of the k-th *rera*?"), which is the one task that legitimately crosses the two timelines: an arousal-indexed anchor and a stage-valued answer. It is instantiated on the sleep-stage grid.

Why different context-length grids. The two regimes operate at qualitatively different time-scales. Arousal events are short (median 19.1 s; class-medians of 18.4 / 20.9 / 18.0 / 15.6 s for the four common classes) and densely interleaved (median inter-event gap 26 s), so a 100 s window already contains a useful retrieval signal while remaining naturally balanced for existence (75% of 100 s windows on the test split contain zero events of a given class). Sleep stages are longer, with the canonical classification length being 30 s. For this reason, the shortest stage grid point is 15 min, as shorter windows would not contain enough stage transitions to populate ordering, antecedent, or multi-hop. We extend stages to 9 h (whole-night) because at this horizon inter-cycle stage ordering becomes the natural retrieval primitive; we cap arousal existence at the shorter horizons because beyond those, *rera* and *hypopnea* saturate existence ($\sim 96\%$ of subjects positive).

Sleep-Haystack: construction pipeline. Unlike Capture24-Haystack we do *not* use needle insertion. Sleep follows highly structured cycles: Wake $\rightarrow$ N1 $\rightarrow$ N2 $\rightarrow$ N3 $\rightarrow$ REM, with REM episodes lengthening across the night (Berry et al., 2017), governed by the AASM scoring rules and the ultradian organisation of human sleep. Splicing a synthetic REM bout into sustained N3, or an obstructive apnea into a Wake epoch, would create physiologically inconsistent segments. Sleep is therefore a good example of the *natural-segment sampling*: we use unmodified slices of the original recordings and let natural variation in window content provide the difficulty gradient.

Construction is again a deterministic function of the (label family, task, context length, sample index, split) tuple via a seeded RNG. The shared procedure is:

1. **Window index.** For each (subject, context length L) we build, once, the index of valid window starts: a sliding window of length L steps across the recording at stride $\min(L/4, 30\,\text{min})$ for sleep stages and $\min(L/4, 5\,\text{min})$ for arousals, and a start is admitted if the window contains at least one bout from the relevant regime. The 9 h length is treated as the whole recording (one window per subject), which may vary in length.
2. **Window draw.** Sample generation draws (subject, window_start) pairs from the per-split slice of this index. The window is presented to the model as starting at $00:00:00$, so all timestamps in question and answer are window-relative.
3. **Window-scoped QA.** The regime timeline is clipped to the window: bouts straddling a window boundary are truncated to the window edge before being indexed.

C.9 ARTS CLASSIFIER CONSTRUCTION (SLEEP)

The Sleep ARTS pre-pass exposes *two* parallel classifier tools to the orchestrator, one per label family. Both share the same backbone: a frozen Chronos-2 multivariate encoder (Ansari et al., 2025) consuming all 13 PSG channels at 100 Hz, feeding a 2-layer MLP head (Linear$(d, 512) \rightarrow$ ReLU $\rightarrow$ Dropout$(0.3) \rightarrow$ Linear$(512, K)$) with K depending on the label family. Training uses AdamW (lr 10^{-3}, weight decay 10^{-4}), batch size 64, class-weighted cross-entropy with weights inversely proportional to per-split class frequencies, cosine LR schedule, and a 30-epoch budget with best-validation-macro-F1 checkpoint selection. The encoder remains frozen throughout; only the head is trained. Both classifiers are trained on the same participant split as the benchmark, so a subject used in any benchmark window is never used to train either classifier.

The two configurations differ in their per-window contract and in how training samples are drawn from each subject:

- **Sleep-stages classifier.** $K = 5$ (*Wake*, *N1*, *N2*, *N3*, *REM*). The deployed inference window is 30 s (3,000 samples), matching the AASM scoring grain of the source annotations. Training samples are the natural deterministic tiling: every contiguous 30 s epoch inside an annotated bout becomes one (subject, start, end, label) example.

- **Arousals classifier.** $K = 6$ (*rera*, *hypopnea*, *obstructive_apnea*, *central_apnea*, *mixed_apnea*, plus a synthetic *none* class). The deployed inference window is 20 s (2,000 samples). Positive examples are one window per kept event with a random offset that guarantees overlap of $\min(10\,\text{s}, \text{event length})$ between window and event. Negative (*none*) examples are drawn from gaps that lie ≥ 10 s away from any annotation, with one negative per positive event per subject; offsets and negatives are re-sampled each epoch via a per-epoch RNG seed so the head sees a fresh slice of the (continuous) window space across training.

Test-set metrics for the deployed checkpoints are reported in Table 10. These classifiers are intended as practical baselines rather than state-of-the-art systems. Future work could explore better performance on agentic retrieval using existing frontier architectures for sleep-stage and arousal classification.

Table 10: ARTS classifier metrics on Sleep. Both classifiers use a frozen Chronos-2 multivariate encoder over the full 13-channel PSG input at 100 Hz with an identical 2-layer MLP head and identical optimizer/schedule.

Classifier	Window	Macro-F1	Accuracy
sleep stages	30 s	0.731	0.746
arousals	20 s	0.464	0.626

Table 11: Per-class F1 for the deployed arousals classifier on the held-out test split.

Class	rera	hypopnea	obstructive_apnea	central_apnea	mixed_apnea	none
F1	0.39	0.47	0.41	0.57	0.12	0.82

C.10 LTAF-HAYSTACK

LTAF: base dataset. We use the PhysioNet Long-Term Atrial Fibrillation database (Petrutiu et al., 2007) in its public release: 84 two-lead ECG recordings sampled at 128 Hz, each approximately 24 hours long, totalling ∼1,961 source-hours. Recordings are annotated against two parallel timelines that we treat as the two label families of LTAF-Haystack.

Rhythm timeline. Bouts are annotated against the AHA rhythm vocabulary, which we restrict to the 9 rhythm codes that occur in the 84-record subset: *NSR* (normal sinus rhythm), *AFIB* (atrial fibrillation), *SBR* (sinus bradycardia), *AB* (atrial bigeminy), *B* (ventricular bigeminy), *T* (ventricular trigeminy), *SVTA* (supraventricular tachyarrhythmia), *VT* (ventricular tachycardia), and *IVR* (idioventricular rhythm). The bout distribution is heavily skewed: *NSR* and *SBR* together account for the majority of annotated time, while *T*, *VT*, and *IVR* are rare (Table 12). This skew is preserved during sample construction, no resampling or class re-weighting is applied at generation time.

We partition the 84 records into train / validation / test at the *record* level with a deterministic seeded shuffle (seed 42, ratios 0.8 / 0.1 / 0.1), yielding 67 / 8 / 9 recordings. All needles, backgrounds, and questions for a given LTAF-Haystack split are drawn exclusively from the corresponding LTAF split, so no recording contributes to more than one split.

LTAF-Haystack: benchmark statistics. LTAF-Haystack instantiates ten TS-Haystack tasks at four context lengths — 100 s, 15 min, 1 h, and 2 h, with a fixed 1,000 / 150 / 150 train / validation / test sample budget per (task, length) cell. Three cells are gated off because the target class saturates: *existence* at 2 h and *anomaly_detection* at 1 h and 2 h, as most LTAF windows contain most of the class labels at these lengths and the yes/no question becomes near-uniformly "yes"). Five 100 s cells under-fill the 1,300 target by a small margin (1,261 to 1,297 samples) because their joint annotation constraints (e.g., a localizable rhythm transition inside a 100 s window) admit fewer feasible windows; thin shards are accepted rather than topped up by synthesis.

LTAF-Haystack: construction pipeline. We use natural segment sampling for LTAF-Haystack. The decision not to use needle insertion is motivated by the idea that ECG carries strong inter-beat

Table 12: Annotated rhythm bout counts in the 84-record LTAF subset (across all splits, after the LTAF timeline builder). Rare rhythms (*T*, *VT*, *IVR*) constrain how many samples can be drawn at the shorter context lengths.

Rhythm	Bouts
NSR	22,879
SBR	11,326
AFIB	7,358
AB	4,472
SVTA	3,268
B	2,696
VT	828
T	785
IVR	137

dependencies that no boundary-smoothing operation can repair. Splicing an AFIB bout into a sinus background would produce a distributional artifact that is not closable by mean alignment or cosine tapering at the seam. Every LTAF-Haystack sample is consequently an *unmodified* two-lead window taken directly from one source recording. The labels driving each task come from the rhythm and beat timelines of the same recording, both of which are intrinsic annotations of that recording's signal.

Sample construction proceeds in three steps and is a deterministic function of the (task, context length, sample index, split) tuple via a seeded random-number generator:

1. **Window indexing.** For each record and each context length L, we enumerate candidate window starts on a stride of $\max(1\,\text{s},\ \min(L/4,\ 30\,\text{min}))$. For every candidate, we precompute (i) a 9-bit rhythm-presence mask (one bit per rhythm in the canonical order) by intersecting the window with the rhythm timeline, and (ii) a 4-tuple of (N, A, V, Q) beat counts from the beat timeline. The result is persisted per (label family, L) as a JSON window index and shared across all tasks at that context length.
2. **Window draw.** For each (task, context length, split) triple, the task generator queries the window index over the split's records using whatever predicate it requires (e.g., "contains ≥ 1 *V* beat" for `anomaly_detection`, or "rhythm r present / absent" for the positive / negative draws of `existence`). The task draws its sample budget by sampling uniformly from the candidate (record, window_start) pairs that satisfy its predicate.
3. **Window-scoped QA.** The task generator clips both the rhythm timeline and the beat timeline to the sampled window, shifts timestamps to be window-relative, and instantiates question and answer from the clipped annotations using the LTAF prompt template bank. The signal is materialized lazily from the per-record `.npy` cache via memory-mapped slicing for efficient training and inference.

C.11 ARTS CLASSIFIER CONSTRUCTION (LTAF)

The ARTS pre-pass on LTAF resembles the Sleep-Haystack setup in that we expose *two* specialized classifiers. A rhythm classifier produces the symbolic timeline used for rhythm-level retrieval, and a beat classifier produces a per-beat label sequence used for beat-level retrieval and anomaly tasks. Both classifiers are trained from scratch on the LTAF train+validation records with the same record-level split as the benchmark and reused unchanged across all LTAF-Haystack tasks.

Rhythm classifier (g_ϕ^{rhythm}). A wide 1D-ResNet (`RhythmResNet1D`: stem plus four stages with two basic residual blocks each, `base_channels` $= 64$, 8.79 M parameters) takes z-scored two-lead 10 s windows at 128 Hz and emits a softmax over the 6-class subset {NSR, AFIB, SBR, AB, SVTA, B}. {IVR, VT and T} rhythm-heads are untrained due to low number of trainable samples. At pre-pass time the classifier is tiled across the QA window in non-overlapping 10 s chunks and consecutive same-label predictions are merged into the derived rhythm timeline. Test-time augmentation with 7 random window-start offsets and softmax averaging ($+4.2$ macro-F1 points over single-view) is the deployed inference setting.

Beat classifier (g_ϕ^{beat}). A History-Time-Frequency (HTF) ensemble (`EcgBeatHTFClassifier`: 1.14 M parameters) operates on 2 s two-lead windows. Three parallel streams capture complementary information: a 1D-CNN over the raw two-lead time window (QRS morphology), a 1D-CNN over the log-magnitude rFFT of the same window (spectral signature of *V* vs. *A* vs. *N*), and an MLP over the $K{=}5$-step history of preceding RR intervals and preceding predicted labels (rhythm context such as bigeminy / trigeminy coupling, teacher-forced at training, autoregressive at inference). Stream features are concatenated before a 3-class MLP head over $\{\text{N, A, V}\}$. The unclassifiable *Q* class is dropped from the head because its support is negligible in this subset ($\sim 0.06\%$ peak per-record ratio; see Appendix C.10).

Test results. The two classifiers are evaluated on the 9 held-out test records (no overlap with training records). Headline figures are summarised in Table 13. We report the rhythm classifier at both single-view and TTA-7 inference, and the beat classifier at its native 2 s window.

Table 13: ARTS pre-pass classifiers on LTAF-Haystack. Both models are trained from scratch on the 67 train and 8 validation records and evaluated on the 9 held-out test records. Macro-F1 is reported alongside accuracy because of LTAF's class imbalance.

Classifier	Setting	Accuracy	Macro-F1
Rhythm (6-class, 10 s)	single-view	0.636	0.614
	TTA-7	0.684	**0.656**
Beat (3-class, 2 s)	single-view	0.954	**0.936**

C.12 Task construction details

In the following, we provide details of each of the 10 proposed tasks for TS-Haystack, exemplified with a random sample visualization from the Capture24 Haystack generation pipeline. Task variations such as *non* Existence, *non* Anomaly Detection or all 4 Comparison variants are excluded for brevity.

Task 1: Existence. *"Is there {activity} in this recording?"*

This task tests whether the model can detect the presence of a specific activity pattern within a longer recording. To prevent variance-based detection shortcuts (where models detect *any* insertion rather than the specific activity), we insert multiple same-regime distractors. The model must learn activity-specific patterns rather than relying on statistical anomalies.

Semi-synthetic regime construction: (1) Sample background window; (2) Select activity regime (sedentary/active); (3) Insert N needles from regime activities not in background; (4) For positive samples, ask about an inserted activity; for negative, ask about a non-inserted activity from the same regime.

Example: *Q: "Is there manual-work in this recording?" A: "No."*

Task 2: Localization. *"When did the {activity} bout occur?"*

Extends existence by requiring temporal grounding. The model must not only detect the activity but report its precise time range. Distractors with similar statistical properties force the model to identify activity-specific temporal signatures.

Semi-synthetic regime construction: Same as Existence, but answer requires the timestamp range of the target needle.

Example: *Q: "When did the sports bout occur?" A: "From 02:09:52:635 to 02:10:08:967."*

Task 3: Counting. *"How many {activity} bouts occurred?"*

Tests enumeration capabilities: the model must detect all instances of an activity and count them correctly. Minimum gaps between bouts ensure distinguishability while making the task non-trivial.

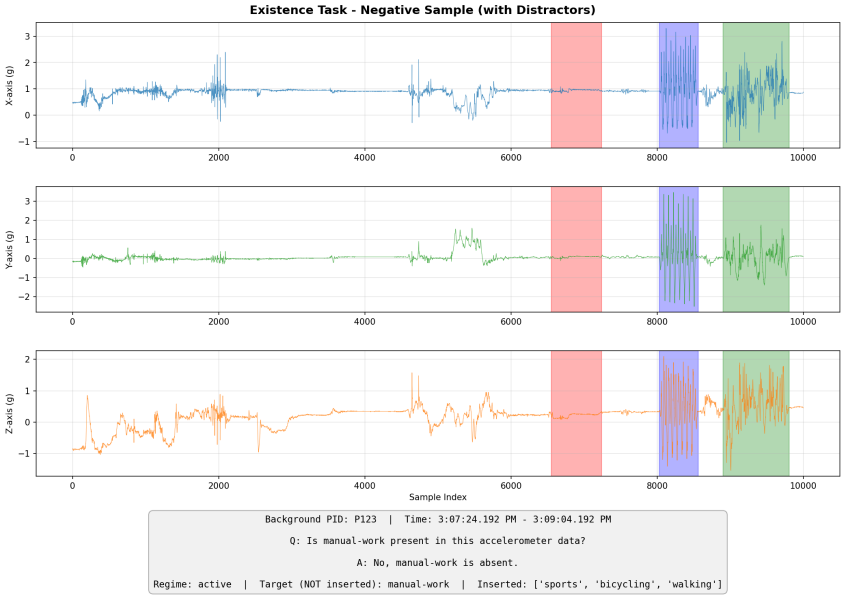


Figure 6: Task 1: Existence. The model must detect whether a specific activity is present among same-regime distractors.

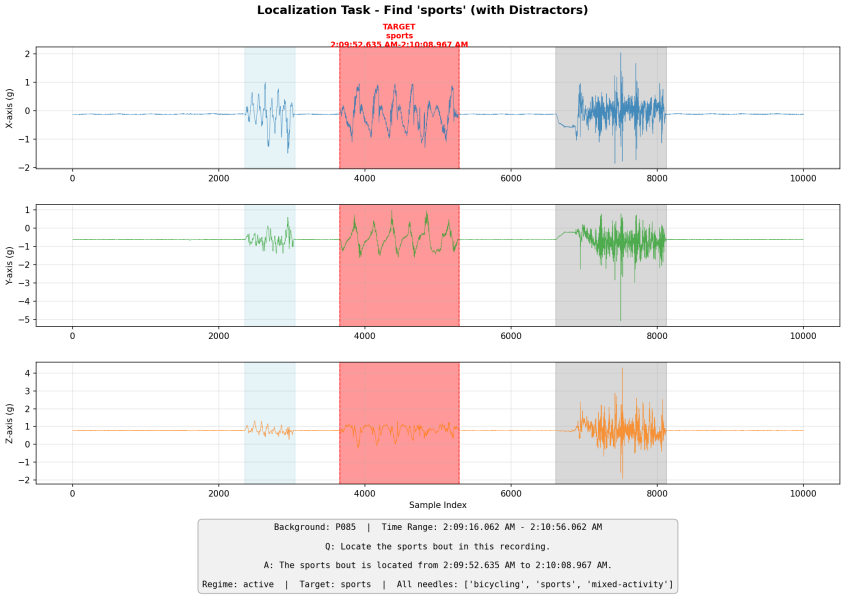


Figure 7: Task 2 (Localization). The model must identify and temporally ground a specific activity

Semi-synthetic regime construction: (1) Sample background; (2) Sample $N \in [1, 5]$ bouts of target activity; (3) Insert at non-overlapping positions with minimum gap constraint.

Example: *Q: "How many standing bouts occurred?" A: "3."*

Task 4: Ordering. *"Did {activity_a} occur before {activity_b}?"*

Tests temporal comparison between two distinct activities. We use needle insertion rather than natural bout selection to prevent models from exploiting learned activity transition statistics. Using random needle sampling pairings instead of existing activity transitions in the data ensures models cannot

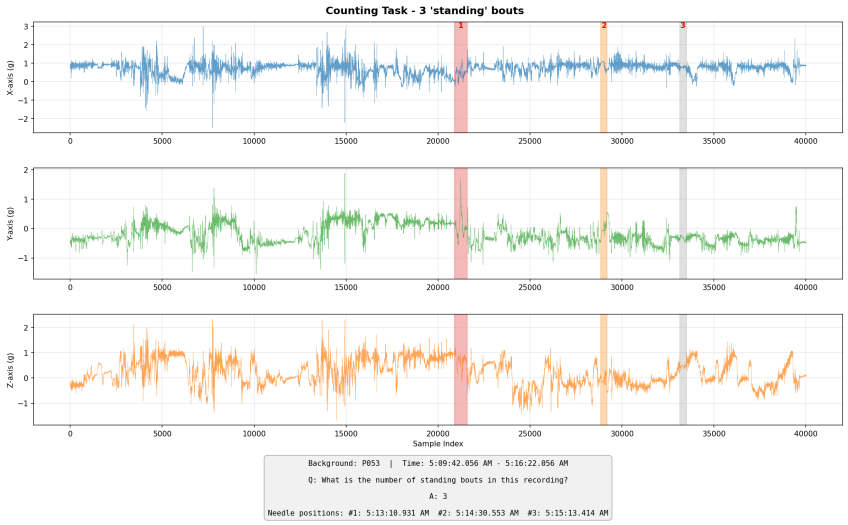


Figure 8: Task 3 (Counting). The model must enumerate all occurrences of a specific activity

game the benchmark through learned Markovian priors, requiring genuine temporal retrieval for correct answers (Chan et al., 2024).

Semi-synthetic regime construction: (1) Sample two distinct activities A and B; (2) Sample background excluding both; (3) Randomly assign temporal order (50/50 balance); (4) Insert both needles sequentially.

Example: *Q: "Did sports occur before sleep?" A: "Yes."*

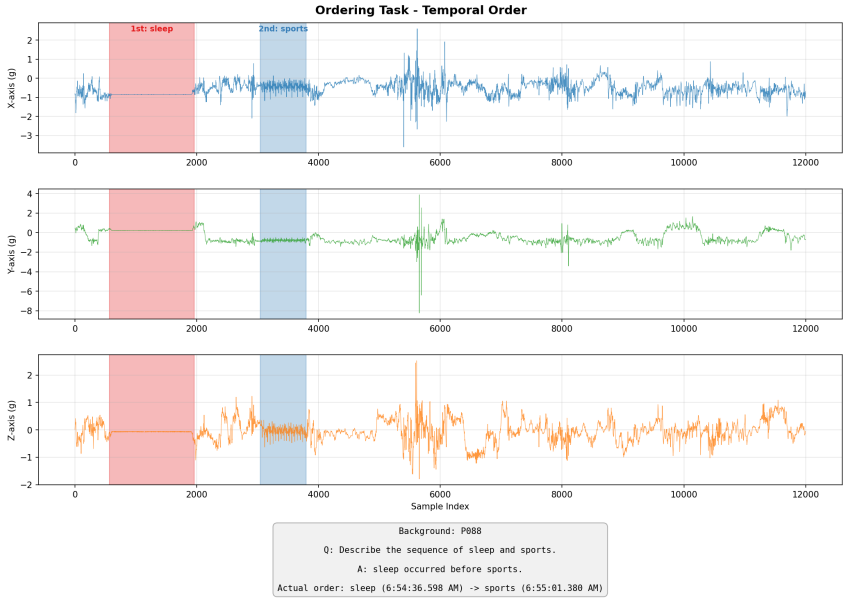


Figure 9: Task 4 (Ordering). The model must determine the temporal relationship between two activities

Task 5: State Query. *"What was the activity level when {event} occurred?"*

Tests cross-scale integration: the model must detect a local event (needle, seconds-scale) and identify the surrounding global activity regime (minutes-scale). This requires understanding hierarchical temporal context.

Semi-synthetic regime construction: (1) Sample mixed background with 2–5 activity states; (2) Select a global state region; (3) Insert needle within that region; (4) Ask about the global state, not the needle activity.

Example: *Q: "What was the overall activity when the vehicle activity occurred?" A: "Standing."*



Figure 10: Task 5 (State Query). The model must identify the global activity regime surrounding a local event

Task 6: Antecedent. *"What activity occurred immediately before {target}?"*

Tests sequential reasoning: the model must locate the target activity and identify what directly preceded it. Two-needle insertion gives control over the antecedent-target pairing, preventing reliance on natural transition statistics.

Semi-synthetic regime construction: (1) Sample background (prefer low-activity); (2) Sample antecedent A and target T; (3) Insert A followed immediately by T with small adjacency gap.

Example: *Q: "Looking at the sequence, what came before bicycling?" A: "Sports."*

Task 7: Comparison. *"What was the {longest/shortest} period {with/without} {activity}?"*

Tests comparison reasoning with four question variants: longest/shortest $\times$ with/without. The "without" polarity requires understanding negation and computing gaps between activity bouts. All inserted bouts have distinct durations to ensure unambiguous answers.

Semi-synthetic regime construction: (1) Sample question type (extremum $\times$ polarity); (2) Insert $N \geq 2$ bouts with distinct durations; (3) For "with": answer is extremum bout; for "without": answer is extremum gap.

Example: *Q: "What was the longest period without household-chores?" A: "From 8:32:52:619 to 08:35:30:682."*

Task 8: Multi-Hop. *"When did the K-th {target} occur {before/after} {anchor}?"*

Tests multi-step retrieval: (1) locate the anchor activity, (2) identify target bouts relative to anchor, (3) count to the K-th occurrence in the specified direction. Optional distractors on the opposite side test directional understanding.

Semi-synthetic regime construction: (1) Sample anchor A and target T; (2) Sample $K \in \{1, 2, 3\}$ and direction; (3) Position anchor with room for K targets; (4) Insert K target needles in order.

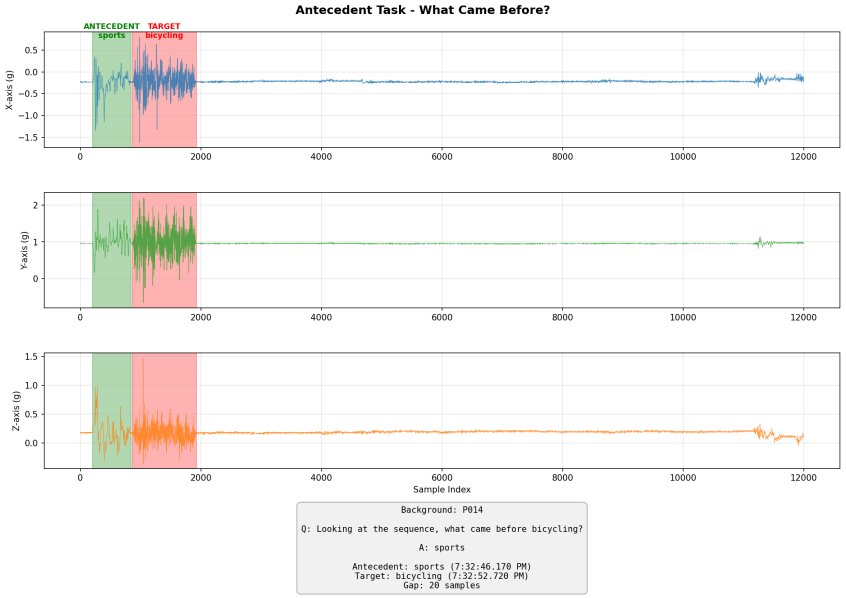


Figure 11: Task 6 (Antecedent). The model must identify the activity immediately preceding a target

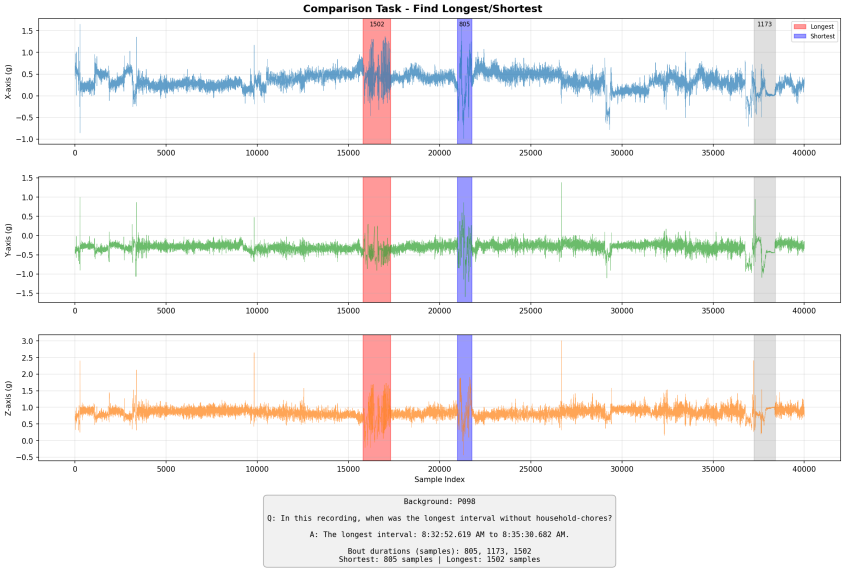


Figure 12: Task 7 (Comparison). The model must compare durations across multiple activity bouts or gaps

Example: *Q: "When did the third sports bout occur after mixed-activity?" A: "From 09:38:32:874 to 09:38:40:924."*

Task 9: Anomaly Detection. *"Is there an anomaly in this recording?"*

Tests anomaly detection for contextual or clinically established anomalies: the model must detect activity that is anomalous *relative to the background regime*. Unlike Existence (where the target is given), the model must identify what constitutes an anomaly. In Capture24, as no natural anomalies are present, we define anomalies as cross-regime insertions (e.g., vigorous activity in a sedentary background). We partition activities into two regimes: *sedentary* (sleep, sitting, standing, vehicle) and *active* (walking, mixed-activity, bicycling, manual-work, sports, household-chores). Mandatory

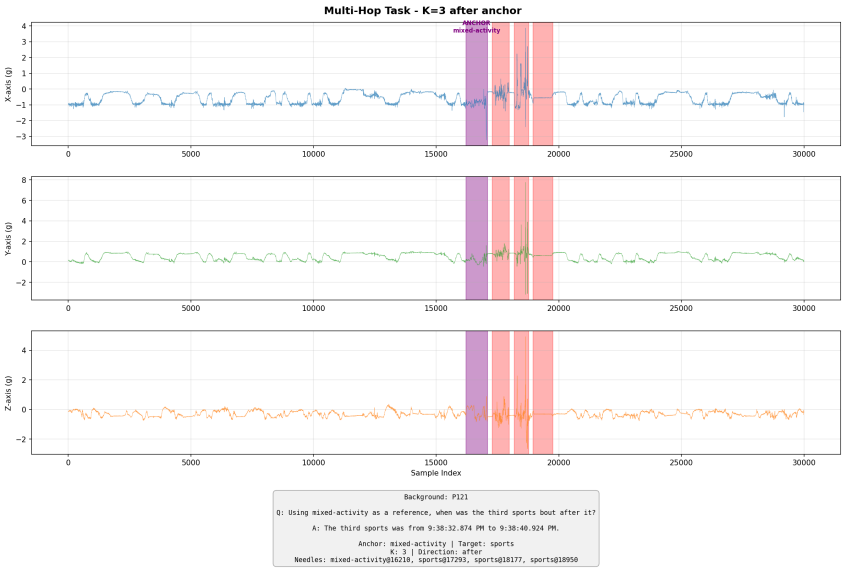


Figure 13: Task 8 (Multi-Hop). The model must locate an anchor, then count to the K-th target in a specified direction

same-regime distractors prevent "insertion detected" shortcuts: both positive and negative samples contain insertions, so the model must identify regime mismatch.

In Sleep and ECG, these anomalies are the clinically meaningful arousals (e.g. (hypo-)apnea, bruxism, ...) and beats (ectopic beats, atrial fibrillation, ...).

Semi-synthetic regime construction: (1) Sample pure background from one regime; (2) Positive: insert 1 cross-regime needle (anomaly) + N same-regime distractors; Negative: insert only same-regime distractors.

Example: *Q: "Is there an anomaly in this recording?" A: "Yes, there is anomalous sitting activity in the active background."*

Task 10: Anomaly Localization. *"Is there an anomaly, and if so, when did it occur?"*

Extends anomaly detection by requiring temporal localization. The model must detect the cross-regime violation and specify its time range. This combines contextual reasoning with precise temporal grounding.

Semi-synthetic regime construction: Same as Anomaly Detection, but positive answers must include the time range of the anomalous activity.

Example: *Q: "Is there an anomaly, and if so, when?" A: "Yes, there is anomalous sitting activity from 04:21:20:792 to 04:21:34:333."*



Figure 14: Task 9 (Anomaly Detection). The model must identify cross-regime activity without being told what to look for

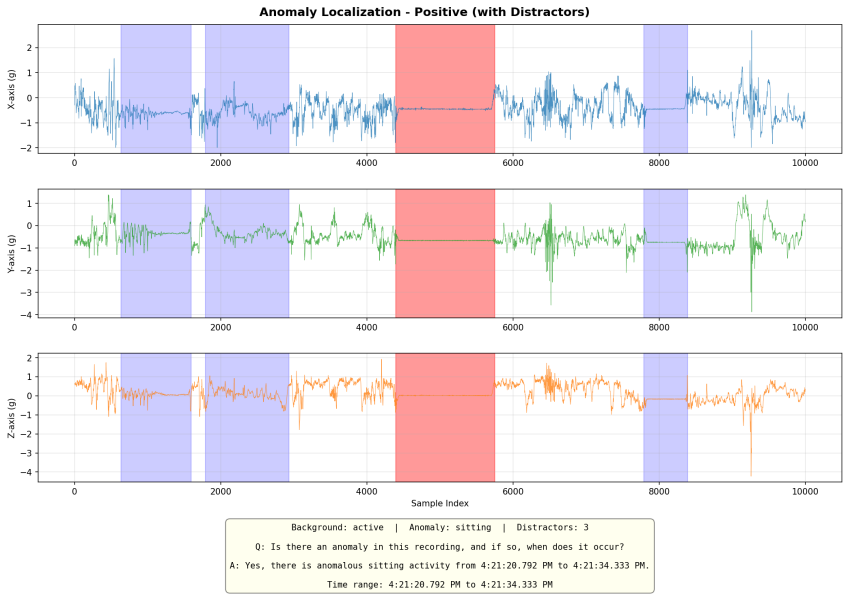


Figure 15: Task 10 (Anomaly Localization). The model must detect and temporally ground cross-regime anomalies