TIME: Temporally Intelligent Meta-reasoning Engine for Chronologically Aware Language Models

Susmit Das[*1]

[1]The Coherence Initiative

October 17, 2025

Abstract

Large language models today exhibit impressive problem-solving skills but remain blind to time, treating interactions in isolation and failing to adapt to elapsed intervals, conversational gaps, or shifting context. We introduce TIME, the Temporally Intelligent Meta-reasoning Engine, a novel behavioral alignment framework that endows LLMs with explicit temporal awareness and meta-cognitive reasoning. Unlike conventional chain-of-thought methods, TIME enables transient, situational meta-reasoning: models use reasoning blocks to reason briefly and purposefully, triggered only by temporal or contextual cues, while structured temporal metadata encodes chronology at every turn. Our multi-phase curriculum and alignment dataset span single-turn, multi-turn, and long-gap dialogues, enriched with explicit "tick" events to model silence and continuity. Applied to Qwen3 models at multiple scales, TIME yields emergent behaviors such as temporal adaptiveness, plan recall, and contextually modulated verbosity, allowing models to decide not only what to say, but when and why, while self-regulating reasoning propensity. We also introduce TIMEBench, a dedicated evaluation suite for temporal reasoning and meta-cognitive adaptation, validating our approach on chronological inference, anomaly detection, and continuity. Our results establish meta-reasoning using temporal and contextual cues as a missing layer for LLMs, and show that frameworks like TIME can encode temporality and meta-cognition in scalable models.

1   Introduction

Large language models (LLMs) still treat dialogue as timeless text. Chain-of-thought (CoT) prompting can *simulate* temporal reasoning—by printing every step—but this brute-force style is verbose, high in latency, and blind to when silence, brevity, or context refresh are appropriate. True **temporal intelligence** demands **meta-reasoning**: an internal controller that decides *whether, when, and how much* to think, triggered by contextual cues.

We present **TIME** (Temporally Intelligent Meta-reasoning Engine), a behavioral alignment framework that turns time into a **first-class control signal**. Each user turn carries an ISO-8601 `<time>` tag; the model may embed short `<think>` blocks anywhere in its reply, or none at all; *tick* turns (time-only inputs) model conversational silence. TIME trains the model to treat these cues as levers for reasoning, tone, and verbosity.

Meta-reasoning here denotes an internal control loop that lets a model decide—on the fly—*whether, when, and where* to think, and *how much* of that thought to reveal. The trigger is not task difficulty alone but **situational cues**: elapsed time, silence, re-engagement, user mood, or discourse shift. Reasoning thus appears in brief, optional bursts embedded anywhere

---

[*]susmitdas@alum.iisc.ac.in

in a turn, or not at all, and may support arithmetic, context recovery, or tone adjustment. In contrast to hybrids that flip a static `/think` switch, meta-reasoning is transient and recursive: the model continuously re-evaluates its own need to reason, producing no trace when none is warranted and inserting concise `<think>` blocks only when temporally or contextually justified.

Because no existing corpus or base model exhibits this behavior, we build a **three-phase curriculum** that first shortens CoT traces, then grafts temporal cues, then teaches reasoning restraint, followed by a **128-sample, full-batch alignment set** whose only invariant is time-aware meta-reasoning. The result is a Qwen-3 variant that decides *what* to say, *why*, and *when*, rather than dumping monologues.

To measure progress we introduce **TimeBench**, 77 scenarios spanning chronology, anomaly, gap awareness, timezone offsets, and adaptivity. Experiments across 4B–32B checkpoints show robust temporal awareness despite using significantly fewer tokens compared to CoT.

Contributions

- **Framework** – A lightweight dialogue protocol (time tags, transient `<think>`, tick turns) that makes time a behavioral driver for LLMs.

- **Curriculum** – Three low-cost scaffold phases plus a full-batch, diversity-max alignment step that isolates temporal meta-reasoning with just 128 examples.

- **Method** – Reproducible QLoRA recipe using limited compute and deterministic checkpoint criterion that generalizes across 32B→4B scales.

- **Benchmark** – TimeBench, seven categories, binary scoring, bootstrap CIs.

- **Results** – Qwen3 models aligned with Time self-regulate reasoning, modulate presentation style and exhibit significant improvement in temporal intelligence over base Qwen3.

2 Related Work

Temporal Cognition and Time-Aware Modeling

Language models have traditionally handled time as either factual metadata or retrieval context. Timestamp-conditioned pretraining [3] and post-hoc evaluation methods such as *TimeShift* [4] analyze the stability of factual recall across time. Systems like TG-LLM [14], GETER [6], and TimeR^4 [10] extend this by integrating temporal graphs or structured temporal retrieval. While effective for explicit date reasoning or event ordering, these approaches focus on the *semantic content* of time, not its *conversational implications*. The model's tone, verbosity, or reasoning mode remains static, regardless of elapsed time or user engagement patterns.

Reasoning Modulation and Self-Regulation

Chain-of-thought prompting [12] improves transparency and accuracy in problem solving, but enforces uniform, turn-global reasoning that bloats output length and suppresses flexibility. Recent models such as Qwen3 and Claude 3.7 allow toggling reasoning via prompt suffixes, while systems like Think-Only-When-Needed [5] introduce binary controllers to automate this switch. More sophisticated methods like DOTS [15], LCPO [1], and Interleaved Reasoning [13] attempt fine-grained control over reasoning quantity and placement—but remain task-bound and formatting-driven, without integrating behavioral context such as timing, fatigue, or continuity.

Curriculum Alignment Strategies

Curriculum learning in LLMs typically targets skill acquisition by sequencing tasks from easy to hard [7, 9]. Most operate at instruction or task granularity and focus on maximizing performance metrics. In contrast, behavioral alignment—especially under compute constraints—has received less attention. Methods such as RLHF rely on extensive human feedback and reward modeling, while controller-based methods assume rigid format templates or modular decoupling.

Our work departs from these by treating curriculum as a tool for shaping latent reasoning structure and behavioral reactivity. Temporal awareness is introduced early through minimal supervision and replayed through later phases to maintain continuity. Final alignment is performed with deterministic full-batch updates over maximally diverse samples, allowing behavioral coherence to emerge from structured gradient pressure on a single behavioral invariant (temporal meta-reasoning), which lets generalization emerge from diversity under constraint rather than dataset scale.

3 Framework
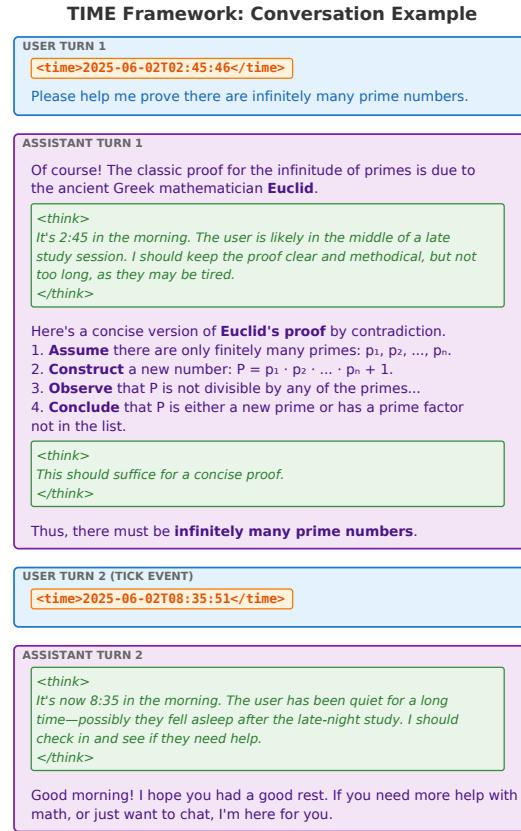
Figure 1: A conversation example in the TIME framework.

Overview

TIME aligns language models to reason *situationally and temporally* by adding three lightweight primitives to the dialogue stream: timestamps, reasoning bursts, and ticks. These structures allow models to condition not just on textual context, but on the passage of time, enabling behavior such as contextual silence, delayed follow-ups, or tone shifts after conversational gaps.

Primitive	Purpose
`<time>`	Absolute timestamp on every turn using ISO-8601 format.
`<think>`	*Optional* short reasoning bursts, inserted anywhere in a response.
Tick	A turn that contains only a `<time>` tag—marks silent time-advances.

Table 1: TIME introduces three core primitives that transform dialogue into a temporally structured behavioral stream.

Message Format

Each user message is prepended with a timestamp, formatted as follows:

```
<time>YYYY-MM-DDTHH:MM:SS</time>
User message
```

This timestamp is **structural metadata**, not just a parsing hint. The model learns to interpret elapsed time as contextual information that informs its behavior—from tone and verbosity to whether it should reason explicitly or remain silent.

Transient Reasoning (`<think>`)

In contrast to classic chain-of-thought, TIME's `<think>` blocks offer a lightweight and localized mechanism for reasoning:

- Reasoning is **optional and context-triggered**, not always invoked.

- Blocks may occur **mid-turn**, anywhere in the utterance.

- Each block is **brief and purpose-driven**, avoiding verbose thought chains.

Tick Events

A **tick** simulates the passage of time in the absence of user interaction. It consists of only a timestamp and no accompanying message:

```
<time>YYYY-MM-DDTHH:MM:SS</time>
```

These tick events give the model a zero-prompt opportunity to decide whether to respond, check in, or remain silent. They occur periodically (e.g., every 30 minutes or every 6 hours) and are key to modeling behaviors like delayed follow-up, plan recall, and long-gap continuity.

Because tick turns impose a strict token budget, the model must make rapid decisions about whether and how to respond—enabling behaviors reminiscent of stateful agents, yet implemented within a stateless transformer framework.

Temporal Behavior as Sequence Control

Together, `<time>`, `<think>`, and ticks transform dialogue into a **temporally anchored behavioral sequence** as documented in Table 1. Rather than treating each exchange as a flat input-output mapping, the model learns to:

- *Reason when needed,*

- *Speak when appropriate,* and

- *Pause when context dictates.*

Figure 1 illustrates a full exchange: a late-night math question, a temporally aware reply with two `<think>` blocks, a silent tick six hours later, and a follow-up that recognizes elapsed time with modulated tone and brevity.

4   Methods

4.1   Curriculum and Alignment Protocol

TIME is aligned through a staged curriculum based supervised fine-tuning process applied to the **Qwen3 family** of open models at **four sizes (32B, 14B, 8B, 4B)**. All training was conducted on a **single RTX 6000 Ada (48GB)** via **QLoRA** [2], under tight compute constraints. The goal is to induce a fundamentally novel behavior—**situational, context-sensitive meta-reasoning**—despite the absence of any pre-trained models or datasets that exhibit such behavior.

Qwen3, though highly capable at reasoning, defaults to a rigid chain-of-thought mode: reasoning is always "on," `<think>` blocks appear monolithically at the top of responses, and verbosity is maximized. TIME requires the opposite: reasoning that is **optional, brief, interruptible, and context-aware**. While Qwen3 does allow non-reasoning outputs via prompt suffixes like `/no_think`, it lacks the ability to self-regulate reasoning deployment dynamically.

Directly overwriting Qwen3's default reasoning was not viable. The target paradigm—*temporally anchored, behaviorally adaptive reasoning*—is absent from existing corpora and cannot be robustly synthesized: even top-tier models like GPT-4o or Gemini 2.5 Flash fail to generate high-quality samples beyond narrow, templated scaffolds. Instead of brute-forcing the target behavior, we designed a scaffolded curriculum, teaching adjacent capabilities—brevity, temporal parsing, and reasoning restraint—in early phases to lay the foundation for a final minimal-data alignment.

All three initial phases share the same training configuration: **LoRA** rank 32 applied to projection layers; $\alpha = 32$; dropout $= 0.05$; ADAMW-8BIT optimizer with learning rate 2×10^{-5}; linear decay schedule; effective batch size 32; gradient checkpointing enabled; and training for 3 epochs. Each phase replays 25% of prior data, and approximately 5% of training samples include brief system prompts describing the TIME framework, phrased with diverse syntax to avoid overfitting.

Synthetic data for the first three phases was generated using a template-guided process seeded by hand-curated examples on GPT-4o and Gemini 2.5 Flash, followed by automated sanity checks. While this data lacked stylistic variety, formatting richness, and topical depth, it was sufficient to teach the adjacent behaviors targeted in each phase: structural parsing, temporal grounding, and reasoning modulation.

Phase 1: Structural Seeding

(2,188 train / 387 test)
This phase introduces the framework's **structured metadata** (i.e., `<time>` tags and `<think>` blocks) in **single-turn settings**, focused on arithmetic and calendric prompts. Each response includes multiple short `<think>` blocks referencing the time or decomposing simple steps.

The goal is not complex reasoning, but to **expose the model to compact, interleaved internal reasoning** and establish a norm of brevity. Despite its simplicity, this phase begins to erode the base model's default long-form CoT habits.

Phase 2: Temporal Exposure

(4,745 train / 838 test → 5,291 / 935 with replay)
This phase expands to **two-turn dialogues**, with the second turn delayed by a **time gap** across a variety of scenarios. The model must now read time metadata and respond accordingly—either by acknowledging the elapsed interval, shifting tone, or invoking memory. **Tick events** (i.e., user turns containing only a `<time>` tag) appear here for the first time.

The key behavior targeted is the explicit internalization of temporal cues—reasoning that reflects elapsed time, inferred user state, or contextual progression. Outputs remain short and stylistically uniform, but establish the foundation for temporal adaptivity. This phase also introduces early pressure against reflexive verbosity, requiring the model to infer when silence is sufficient or when a concise contextual update is more appropriate.

Phase 3: Contextual Modulation

(4,147 train / 732 test → 5,878 / 1,039 with replay)
Multi-turn (3–5 turn) conversations are introduced, spanning broader topics. Now, the target is modulation of both internal reasoning and external presentation: the model should suppress unnecessary thinking and adjust user-facing tone and verbosity. **Tick events** appear more frequently ($\approx$33% of samples), though overt temporal anchoring recedes.

Replies are shorter and less diverse than ideal, but this phase encodes the **capacity for reasoning suppression and selective deployment**, creating the behavioral room needed for Phase 4. With consistent replay from earlier phases, temporal behaviors remain latent and recoverable, even when stylistically buried.

Together, these phases do not express the full TIME behavior—but they build a behavioral substrate: a model that can read structured time cues, engage or suppress reasoning, and respond to context with emerging modulation of verbosity and internal thought.

Phase 4: Gradient-Aligned Convergence via Maximal Diversity

(128 hand-curated conversations · no replay)

Aspect	Variation Injected
Topics	Speculative fiction, late-night coding, therapy, anime debates, hostile rants, professional emails, gaming chat, etc.
Styles	Terse ↔ verbose, academic markdown ↔ free-form prose, emoji-laden ↔ plain text, bullets, numbered steps, code fences—often flipped mid-dialogue.
Turns	1- to 7-turn threads, with context changes, style switches, and tick events after long gaps.
Noise	Typos, style switches, abrupt topic shifts, contradictory instructions, diverging behaviors.

Table 2: Surface diversity in Phase 4 ensures that only the intended behavioral invariant aligns gradients across the batch.

Phase 4 delivers the decisive alignment. Instead of scaling data, we curate a **small, hand-curated, ultra-diverse batch** whose *only* commonality is the target behavior—contextual meta-reasoning keyed to `<time>` cues. Everything else is randomized as shown in Table 2.

With every surface regularity broken, **diversity itself becomes the filter**: any gradient update that depends on topic or format cancels out, leaving only the temporal-meta-reasoning signal as the *path of least resistance*.

Setting	Value
Effective Batch size	128 (entire dataset)
Optimizer	AdamW-8bit
LR / warm-up	1.5×10^{-4}, 6 warm-up steps
Epochs scanned	30 / 36 / 43 / 46 (32B / 14B / 8B / 4B)
Checkpoint criterion	First epoch with loss < 1.05 and no sharp rise in degeneracy
LoRA	Rank 32 adapters, $\alpha = 32$

Table 3: Final tuning setup for Phase 4 alignment. Clean convergence is assessed based on degeneracy suppression.

Deterministic Full-Batch Fine-Tuning

Training is performed with a batch size equal to the entire dataset: all 128 samples are used in every update step. This full-batch setup is deterministic in inputs, eliminates sampling variance, and ensures the optimizer is exposed to the full entropy of the behavioral landscape at once. Details are mentioned in Table 3.

Across all four scales, we observe an **abrupt degeneracy tipping-point**: once training loss dips below ≈ 1.04, the frequency of infinite loops, <think> format bleed, and style collapse rises sharply. We therefore retain the **last clean epoch**—19 / 25 / 32 / 38 for the 32B / 14B / 8B / 4B models, respectively—corresponding to loss $\in [1.045, 1.050]$. The consistency of this threshold across architectures reinforces the notion that the model converges to a single **gradient-aligned behavioral attractor** and collapses once that attractor is over-optimised.

Why It Works

Processing the entire entropy each step forces every contradictory signal to null out. The **sole coherent gradient** is the behavior present across the dataset:

- Detect elapsed time and tick events.

- Decide *whether* to reveal thought, *how much*, and *where* in the turn.

- Use temporal cues to contextualize the user prompt as well as modulate tone and brevity.

Monolithic CoT, default verbosity, and rigid /think toggles become anti-correlated noise and are pruned away. Phase 4 thus **locks in** the behavior scaffolded by Phases 1–3 without memorizing any single format, yielding the final checkpoints used for evaluation, henceforth referred to as **TIME-32B / 14B / 8B / 4B**.

4.2 Evaluation Method: TIMEBench

TIMEBench is a 77-scenario benchmark purpose-built to evaluate **temporal intelligence** in dialogue models. Scenarios resemble real-world usage and have **no overlap with training data**. The benchmark covers **seven diagnostic categories**, each with 11 scenarios probing a distinct subskill.

- **Chronological Retrospection**[1]: Reconstructs timeline across many turns. *e.g.*, "How long have we been talking?"

- **Invalid Time Detection**[1]: Identifies nonsensical or invalid timestamps. *e.g.*, "Feb 29, 2022" or "25:41"

- **Temporal Adaptivity**[2]: Modulates tone, urgency, or verbosity as deadlines approach or pass. *e.g.*, Being terse when a paper is due in 2 minutes.

- **Temporal Contextual Awareness**[2]: Infers user context or situation from a single timestamp. *e.g.*, Recognizing probable contexts from a 2AM stressed query.

- **Temporal Flow Anomaly Detection**[1]: Detects implausible jumps (>100 years) or reversals in time. *e.g.*, A turn in 2024 followed by one in 2148.

- **Time Gap Awareness**[2]: Adjusts assumptions after plausible silences. *e.g.*, No message from May to July.

- **Timezone Sensitivity**[1]: Detects changes in timezone offset across turns to infer transitions. *e.g.*, UTC+5 to UTC–8 implies travel.

Protocol

- **Binary scoring**: Each model output is judged against a clear objective (score = 0 or 1).

- **Three runs**: Each scenario is run thrice using PCG64-derived seeds from a master seed (total: 231 runs).

- **Blind LLM as judge**: All evaluations are conducted by a third-party LLM, which sees only the response and the objective—not the actual prompt or timestamp—to avoid evaluation bias..

- **Aggregation:** Seed-level scores $\rightarrow$ scenario score (mean of 3) $\rightarrow$ category score (mean of scenarios, scaled to %) $\rightarrow$ overall TIMEBench score (mean of category scores).

TIMEBench differentiates behaviors explicitly trained from capabilities that generalize out-of-distribution, ensuring that scores are not artifacts of memorized scaffolds or surface prompts.

We also capture **95% confidence intervals** via **stratified bootstrapping** (10,000 resamples):

- Resample scenario scores *within each category.*

- Recompute category and overall percentages.

- CI = 2.5^{th}–97.5^{th} percentiles.

5 Results

We evaluate all models on **TIMEBench**, comparing our aligned models (**TIME-4B / 8B / 14B / 32B**) against the base Qwen3 in both the default reasoning mode and the suppressed mode (via the `/no_think` suffix). All models were evaluated using identical sampling parameters: **temperature 0.6**, **top-p 0.95**, **top-k 20**, **min-p 0**, following best practices for reasoning evaluations as recommended in the *Qwen3 Technical Report* [11]. Qwen3 models have been tested with both Thinking (reasoning) and No Thinking (no reasoning) modes.

All evaluations were conducted in **FP8** with dynamic activation scaling, using NVIDIA's 8-bit floating point implementation [8]. This mirrors real-world deployment constraints for large

---

[1]Category is completely out-of-distribution relative to training.

[2]Category is derived from curriculum-intended behavior.

Size	Qwen3		TIME
	No Thinking	Thinking	
4B	21.64	31.17	**51.95**
8B	29.00	38.09	**58.87**
14B	28.57	37.23	**58.01**
32B	31.60	49.78	**66.23**

Table 4: TIMEBench scores (out of 100) across models.
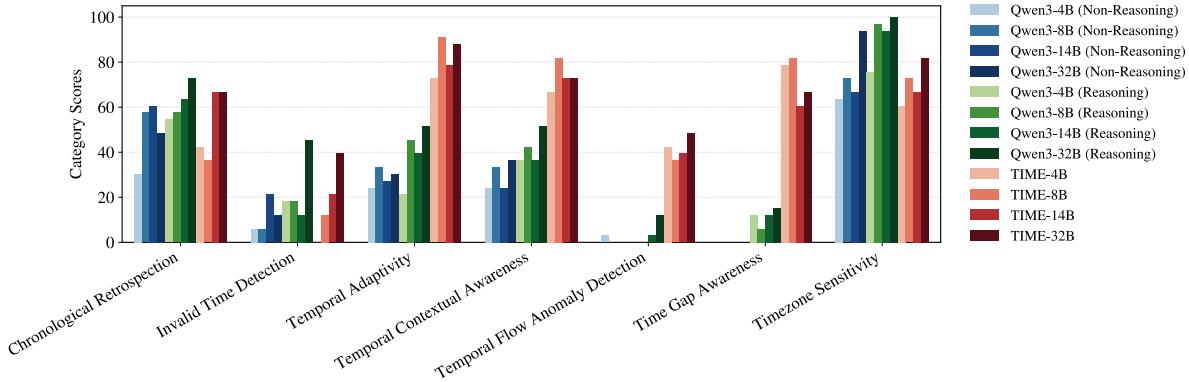
Figure 2: Breakdown of TIMEBench scores by diagnostic category and model size. TIME models consistently outperform both reasoning (Thinking Mode) and non-reasoning (No Thinking Mode) Qwen3 baselines.

models like 32B on a single RTX 6000 Ada GPU. A **BF16-vs-FP8 ablation** in Section 5.2 confirms it doesn't materially affect evaluation results.

Table 4 gives **TIMEBench scores (out of 100)**. As shown in Figure 2, TIME models consistently outperform both reasoning and non-reasoning baselines across most diagnostic categories at all scales. Its 95% confidence intervals obtained from stratified bootstrapping indicate robust gains: for TIME-4B, the interval spans 45.02 to 58.44; for TIME-8B, 51.95 to 65.37; for TIME-14B, 50.65 to 65.80; and for TIME-32B, 58.44 to 73.16. By contrast, even in the reasoning modes, Qwen3 variants fall well below: Qwen3-4B ranges from 23.81 to 38.10, Qwen3-8B from 31.60 to 44.59, Qwen3-14B from 30.74 to 44.16, and Qwen3-32B from 42.42 to 57.14.

To verify the robustness of these gains, we conducted a **Wilcoxon signed-rank test** between per-scenario average scores for TIME and Qwen3 (Reasoning). The improvements were statistically significant across all model sizes ($p < 0.01$): for 4B, $p = 0.0024$; for 8B, $p = 0.0025$; for 14B, $p = 0.0011$; and for 32B, $p = 0.0095$.

This confirms that TIME's temporal reasoning advantages are not artifacts of format, category, or specific prompt design.

5.1 Phase-wise Ablation in 32B (Structural/Behavioral Metrics)

Table 5 traces how each curriculum stage reshapes the **32B** model's surface behaviour and TIMEBench score, while Figure 3 shows how specific temporal competencies evolve across phases using category scores.

Interpretation.

- **Reasoning placement** shifts from always-at-top (100% "Start" in Qwen3) to predominantly mid-turn (≈69%) in TIME-32B, along with selective usage of `<think>` blocks

Model	Score	Runs with <think> (%)	Markdown		Think Position			Tokens (Mean)		Runs with
			Light (%)	Heavy (%)	Start (%)	Mid (%)	End (%)	Only Reasoning	Full Output	Format Bleed (%)
Qwen3 (No Thinking)	31.60	100.0	97.8	86.2	100.0	0.0	0.0	0	635	0.00
Qwen3 (Thinking)	49.78	100.0	98.7	91.3	100.0	0.0	0.0	942	1601	17.32
Phase 1	48.05	99.6	97.8	86.6	100.0	0.0	0.0	809	1468	16.45
Phase 2	58.87	91.8	82.3	65.4	57.3	42.3	0.4	74	450	6.93
Phase 3	48.49	77.5	83.1	68.4	48.3	51.7	0.0	43	322	2.60
TIME	**66.23**	79.2	86.2	64.1	30.9	68.7	0.4	70	353	3.46

Table 5: Phase-wise ablation: Structural and behavioral metrics for Qwen3-32B through curriculum stages and final alignment.

(≈79% runs having thinking blocks compared to 100%), matching our goal of interleaved, context-triggered thinking.

- **Reasoning volume** collapses from ∼1k tokens in vanilla CoT to ∼70 tokens in TIME, cutting generation cost by an order of magnitude.

- **Format bleed**—unwanted leakage of <think> text or markdown artifacts falls from 17% to under 4%, reflecting cleaner delimitation.

- **Markdown adaptivity**—heavy structures (headers, lists, code, tables) appear only when warranted, otherwise the model sticks to light **bold**/*italic*.

- **Figure 3** confirms this progression: Qwen3 over-specializes in timezone handling but lags elsewhere; Phase 2 elevates adaptivity and flow detection; only TIME achieves balance across all metrics.

- Phase 2's score spike (58.9) coincides with strong temporal anchoring but still front-loaded reasoning; Phase 3 teaches suppression, sacrificing score but quashing verbosity; Phase 4 (TIME) restores competence *without* regressing into verbose CoT, yielding the best score (66.2).

Together, these metrics show that TIME's improvement is not a mere accuracy bump; it reflects a **structural transition** from rigid, monolithic reasoning to *situational, economical, and well-bounded meta-reasoning*—the precise behaviour the curriculum was designed to produce.

5.2 Ablation: FP8 vs. BF16

As mentioned earlier, all evaluations were conducted in **FP8 with dynamic activation scaling**. To verify behavioral parity, we re-ran **TIME-8B** and **TIME-14B** checkpoints in **BF16**, using identical sampling parameters: temperature 0.6, top-p 0.95, top-k 20, min-p 0.

As we can see from Table 6, no statistically significant differences were found in reasoning structure, markdown usage, or performance. TIME models exhibit **robust inference behavior under both FP8 and BF16**.

6 Discussion

TIME demonstrates that *temporal awareness is not an emergent side-effect of scale or chain-of-thought prompting*, but a behavior that must be explicitly *taught*. With only ∼11k samples and training on a single GPU, we transform Qwen3 from a monolithic reasoner into a model that decides *when* to think, *how much*, and *why*, while improving general temporal intelligence significantly. The curriculum reveals a coherent trajectory: Phase 1 compresses chain-of-thought
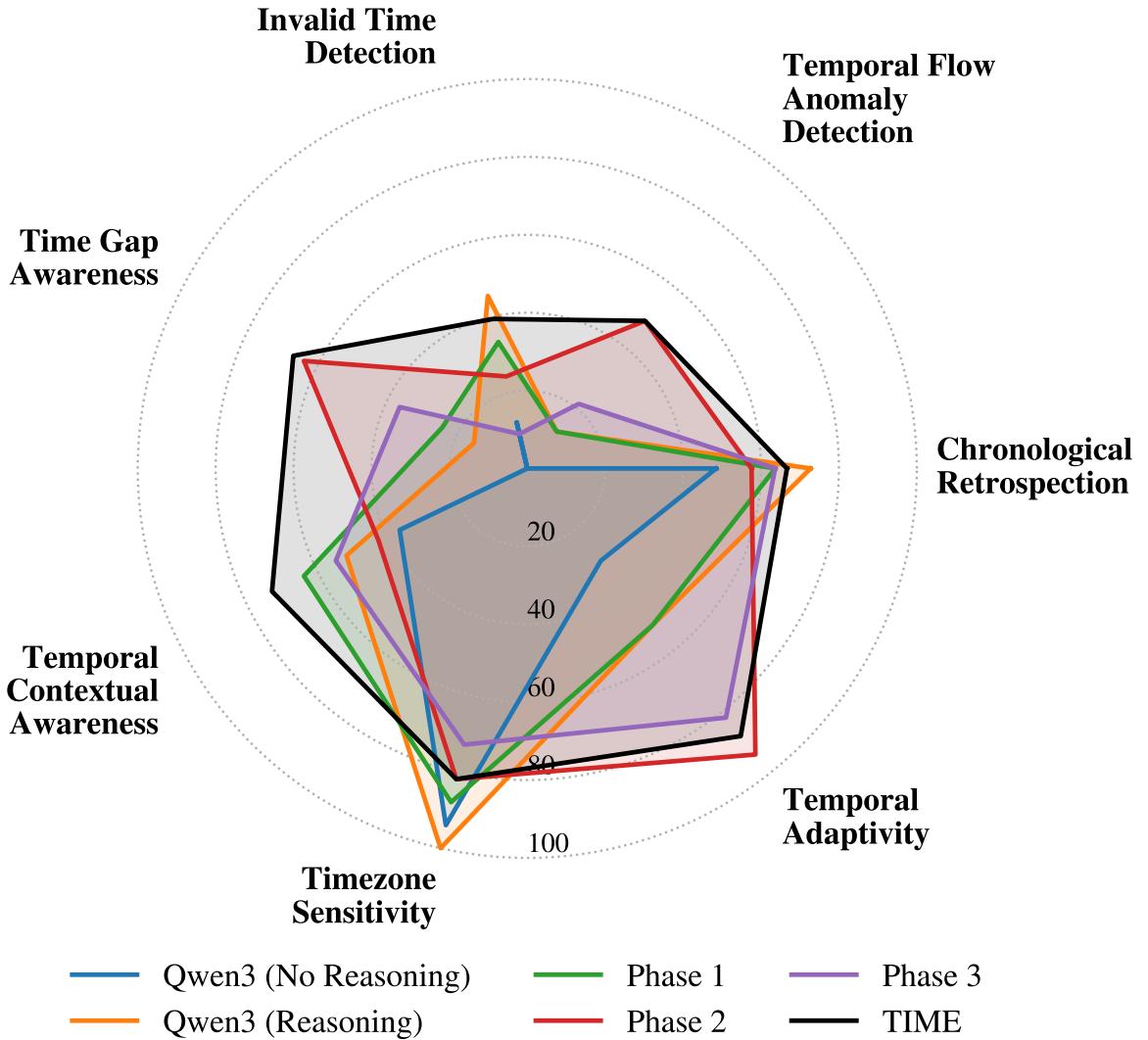
Figure 3: Evolution of core temporal competencies in Qwen3-32B across curriculum phases. Only TIME achieves balanced gains across all diagnostic categories.

traces; Phase 2 introduces temporal cues but remains verbose; Phase 3 teaches restraint; and Phase 4, via full-batch alignment over maximal diversity, locks in the core invariant—*time-conditioned meta-reasoning*—resulting in the balanced radar profile seen in Figure 3.

The method has limitations. Explicit timestamps are required at inference time; without them, the model reverts to generic dialogue behavior. Phase 4 depends on hand-curated data, and while the loss-threshold checkpointing rule is deterministic and repeatable, it remains empirically tuned. TIMEBench exposes significant room for improvement, especially in chronological retrospection and invalid time detection. Moreover, while temporally competent, the models are not encyclopedic and may still fail on obscure calendrical or cultural edge cases. In tasks that reward step by step reasoning, chain-of-thought will still provide immense value despite the computational costs.

These constraints point toward several next steps: generating alignment sets with automated entropy maximization; augmenting with retrieval-based timestamp injection or explicit temporal embeddings; experimenting on building on this paradigm with reinforcement learning based

Metric	TIME-8B		p-value	TIME-14B		p-value
	FP8	BF16		FP8	BF16	
Benchmark score	58.87	56.28	0.40	58.01	59.74	0.91
Heavy markdown usage (%)	67.97	71.43	0.33	62.77	61.90	0.91
Light markdown usage (%)	86.58	87.45	0.51	89.61	87.45	0.38
% of runs with `<think>` blocks	82.68	80.52	0.60	86.58	85.71	0.45
Mean number of `<think>` blocks per run	1.43	1.29	0.36	1.73	1.54	0.89
Mean reasoning tokens per run	70.29	64.28	0.19	87.00	83.38	0.90

Table 6: FP8 vs. BF16 ablation for key metrics in TIME-8B and TIME-14B with p-values from Wilcoxon signed-rank tests on scenario-level aggregations.

methods; and generalizing the gradient-attractor strategy to other meta-reasoning dimensions, such as spatial reasoning, stylistic modulation, or task-phase awareness.

7 Conclusion

We introduce TIME, a lightweight curriculum and alignment framework that equips LLMs with *situational, temporally grounded meta-reasoning* while remaining deployable in FP8. Across four model sizes, TIME lifts temporal accuracy, reduces reasoning token counts by an order of magnitude, and shifts internal thought placement from rigid preambles to brief, context-sensitive interjections. Crucially, the method succeeds because *diversity becomes the filter*: in an alignment batch where every superficial signal conflicts, the sole viable gradient path is to learn temporal meta-reasoning as the common behavioral invariant.

We release TIMEBench, the alignment code and datasets to facilitate future work on models that not only know *what* and *how*, but also *when*.

Reproducibility Infrastructure

All experiments were conducted on a single workstation equipped with an *AMD Ryzen 9 7950X3D* CPU, 128 GB of DDR5 RAM, and a single *NVIDIA RTX 6000 Ada* GPU (48 GB), running *Ubuntu 22.04* via WSL2 (*Windows 11, build 26100*). The system used *NVIDIA driver version 576.02* with *CUDA 12.9*. No multi-GPU, distributed, or cloud-based compute was used at any stage.

The training pipeline was implemented using UNSLOTH (v2025.5.7), TRANSFORMERS (v4.51.3), ACCELERATE (v1.7.0), NUMPY (v1.26.4), llmcompressor (v0.6.0), and SCIPY (v1.15.3) for statistical analysis. Inference runs were conducted using vLLM (v0.10.0) under identical decoding configurations across all checkpoints.

References

[1] Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning, 2025.

[2] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: efficient finetuning of quantized llms. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[3] Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273, 2022. doi: 10.1162/tacl_a_00459. URL https://aclanthology.org/2022.tacl-1.15.

[4] David Herel, Vojtech Bartek, Jiri Jirak, and Tomas Mikolov. Time awareness in large language models: Benchmarking fact recall across time, 2025.

[5] Lingjie Jiang, Xun Wu, Shaohan Huang, Qingxiu Dong, Zewen Chi, Li Dong, Xingxing Zhang, Tengchao Lv, Lei Cui, and Furu Wei. Think only when you need with large hybrid-reasoning models, 2025.

[6] Zihao Jiang, Ben Liu, Miao Peng, Wenjie Xu, Yao Xiao, Zhenyan Shan, and Min Peng. Towards explainable temporal reasoning in large language models: A structure-aware generative framework, 2025.

[7] Bruce W. Lee, Hyunsoo Cho, and Kang Min Yoo. Instruction tuning with human curriculum. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1273–1287, Mexico City, Mexico, June 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.findings-naacl.82.

[8] Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, Naveen Mellempudi, Stuart Oberman, Mohammad Shoeybi, Michael Siu, and Hao Wu. Fp8 formats for deep learning, 2022. URL https://arxiv.org/abs/2209.05433.

[9] Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, and Shuiwang Ji. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning, 2025.

[10] Xinying Qian, Ying Zhang, Yu Zhao, Baohang Zhou, Xuhui Sui, Li Zhang, and Kehui Song. TimeR4 : Time-aware retrieval-augmented large language models for temporal knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6942–6952, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-main.394.

[11] Qwen-Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[12] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022.

[13] Roy Xie, David Qiu, Deepak Gopinath, Dong Lin, Yanchao Sun, Chong Wang, Saloni Potdar, and Bhuwan Dhingra. Interleaved reasoning for large language models via reinforcement learning, 2025.

[14] Zidi Xiong and other authors. Large language models can learn temporal reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.563.

[15] Murong Yue, Wenlin Yao, Haitao Mi, Dian Yu, Ziyu Yao, and Dong Yu. Dots: Learning to reason dynamically in llms via optimal reasoning trajectories search, 2024.

8 Appendix

This appendix provides comprehensive supplementary material supporting the claims and results presented in the main paper. It is organized into four sections:

- **Appendix A**: **Training Details**
  Describes curriculum construction, dataset statistics, and fine-tuning configurations across all phases. Includes sequence length distributions, model adaptation setup (QLoRA), optimizer settings, and the checkpoint selection heuristic used across model sizes.

- **Appendix B**: **Evaluation Methodology**
  Covers the procedural specifics of the **TIMEBench** benchmark—its scoring pipeline, sampling configuration, and evaluation model setup. Also includes implementation details for structural behavior analysis, reasoning token estimation, markdown formatting assessment, and confidence interval computation via stratified bootstrap.

- **Appendix C**: **Detailed Ablations and Metrics**
  Presents a battery of diagnostic metrics—both descriptive and bootstrapped—across model sizes and training variants (e.g., with and without TIME curriculum). Includes Wilcoxon Signed-Rank (WSR) tests at the **scenario level**, comparing TIME-tuned models to baselines. Also features a brief FP8 vs BF16 ablation.

- **Appendix D**: **Representative Outputs**
  Provides curated examples of model behavior:

  - **D.1**: Full conversations with **TIME-32B** on novel temporal prompts (not seen during training or evaluation), illustrating context-sensitive, structured, and economical reasoning.
  - **D.2**: High-scoring completions from **TIMEBench**, with two examples per model size. Only the **final turn** is generated; all prior turns are authored as part of the scenario.

While the appendix sections are self-contained, readers may find it useful to begin with **Appendix D** to build intuitive familiarity with the model's reasoning behaviors before diving into the training and evaluation specifics.

Appendix A.0: Reproducibility Infrastructure

All training, statistical analysis, and inference were performed on a **single machine** with the following hardware and system configuration:

- **CPU**: AMD Ryzen 9 7950X3D

- **Memory**: 128 GB DDR5

- **GPU**: NVIDIA RTX 6000 Ada Generation (48 GB VRAM)

- **Operating System**: Ubuntu 22.04 LTS (inside **WSL2**, Windows 11 Build 26100)

- **CUDA Version**: 12.9

- **Driver Version**: NVIDIA 576.02

- **NVIDIA-SMI**: 550.54.15

- **Compute Capability**: ≥ 8.9 (required for FP8 formats like E4M3)

All experiments were conducted on **single-node, single-GPU** setups without distributed training.

Python Environments

Two separate environments were used due to package compatibility requirements:

A.0.1 Training & Statistical Analysis

Used for all fine-tuning phases, ablations, and post-evaluation analyses:

```
torch==2.7.1
unsloth==2025.5.7
transformers==4.51.3
accelerate==1.7.0
numpy==1.26.4
llmcompressor==0.6.0
scipy==1.15.3
```
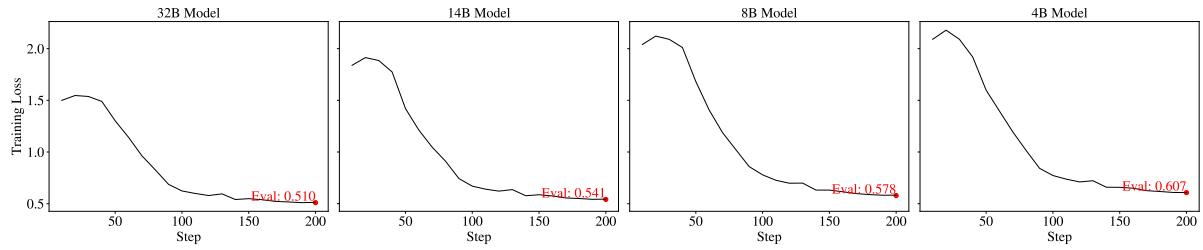
A.0.2 Inference (vLLM Hosting)

Used for all inference-time runs, including TIMEBench:

```
torch==2.7.1
vllm==0.10.0
transformers==4.54.0
```

Each environment was version-locked and isolated to ensure **reproduction** and **non-interference** between toolchains (e.g., Unsloth-based fine-tuning vs. vLLM inference streaming).

Appendix A.1: Phase 1 − Training Configuration

Dataset Size

- **Train set**: 2,188 samples
- **Test set**: 387 samples

Sequence Statistics

- **Train sequences**:
    - Max length: 2,057 tokens
    - Mean: 310.8 tokens
    - 90th percentile: 527 tokens
- **Test sequences**:
    - Max length: 1,504 tokens
    - Mean: 311.1 tokens
    - 90th percentile: 544 tokens
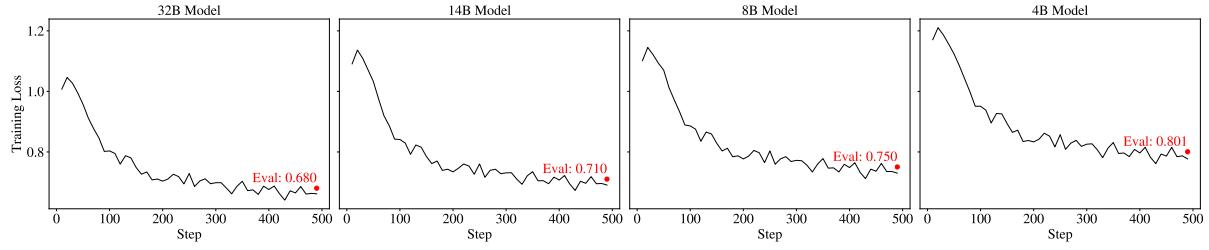
Model & Optimization Details

- LoRA-based fine-tuning on a pretrained LLM using **4-bit QLoRA** with **LoRA rank 32**, alpha 32, dropout 0.05
- Target modules included attention and MLP projection layers (`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`)
- Gradient checkpointing enabled
- Full model not fine-tuned (PEFT setup)

Training Configuration

- Optimizer: `adamw_8bit`
- Effective batch size: 32 (2 × 16 gradient accumulation)
- Epochs: 3
- LR: 2e-5, with linear scheduler and 100 warmup steps
- Max grad norm: 1.0
- Weight decay: 0.01
- Seeded runs for determinism
- Logging: every 10 steps
- Evaluation used the held-out test set (387 samples)

Appendix A.2: Phase 2 – Training Configuration

Dataset Size

- **Initial Train set**: 4,745 samples
- **Initial Test set**: 838 samples
- **After replay integration**:
    - **Final Train set**: 5,291 samples
    - **Final Test set**: 935 samples

Sequence Statistics

- **Train sequences**:
    - Max length: 3,795 tokens
    - Mean: 483.4 tokens
    - 90th percentile: 904 tokens
- **Test sequences**:
    - Max length: 2,548 tokens
    - Mean: 490.3 tokens
    - 90th percentile: 903 tokens
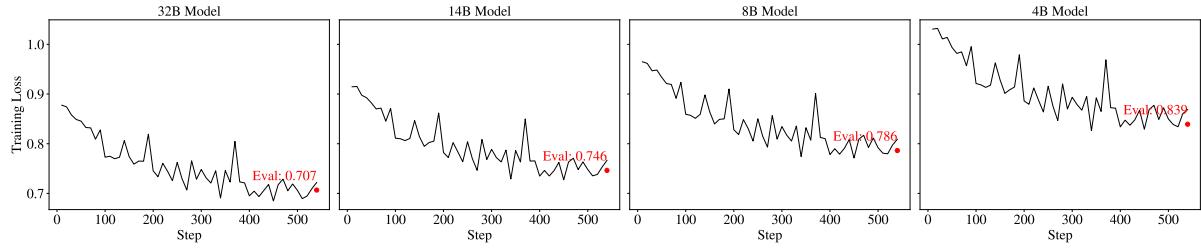
Model & Optimization Details

- Identical fine-tuning stack as Phase 1, leveraging **QLoRA with LoRA rank 32**
- Attention and MLP projections targeted for adaptation
- Full model kept frozen; training restricted to adapter layers
- **Gradient checkpointing** enabled for memory efficiency

Training Configuration

- Optimizer: `adamw_8bit`
- Effective batch size: 32 (2×16 gradient accumulation)
- Epochs: 3
- Learning rate: 2e-5 (linear scheduler with 100 warmup steps)
- Max grad norm: 1.0
- Weight decay: 0.01
- Logging every 10 steps
- Evaluation performed on the updated 935-sample test set
- Deterministic seed used for all training and evaluation

Appendix A.3: Phase 3 – Training Configuration

Dataset Size

- **Initial Train set**: 4,147 samples
- **Initial Test set**: 732 samples
- **After replay integration**:
  - **Final Train set**: 5,878 samples
  - **Final Test set**: 1,039 samples

Sequence Statistics

- **Train sequences**:
  - Max length: 3,795 tokens
  - Mean: 496.0 tokens
  - 90th percentile: 855 tokens
- **Test sequences**:
  - Max length: 3,041 tokens
 - Mean: 464.7 tokens
  - 90th percentile: 752 tokens
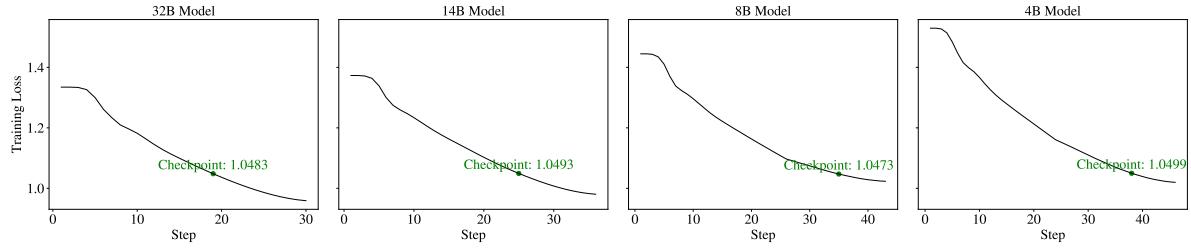
Model & Optimization Details

- Same LoRA configuration as previous phases with $\mathbf{r = 32}$, targeting standard attention and MLP projections
- **QLoRA** used with 4-bit quantization
- **Gradient checkpointing** enabled
- Full model frozen during tuning; only adapter layers trained

Training Configuration

- Optimizer: `adamw_8bit`
- Effective batch size: 32 (2×16 gradient accumulation)
- Epochs: 3
- Learning rate: 2e-5 (with linear scheduler and 100 warmup steps)
- Max gradient norm: 1.0
- Weight decay: 0.01
- Logging every 10 steps
- Evaluation performed on 1,039-sample test set after replay augmentation
- Fixed random seed for reproducibility

Appendix A.4: Phase 4 – Training Configuration

Dataset Size

- **Train set**: 128 hand-curated conversations
- No formal test set; checkpoint selection guided by loss band

Sequence Statistics

- Max sequence length: 1,379 tokens
- Mean: 735.8 tokens
- 90th percentile: 981 tokens

Training Configuration

- Finetuning performed via **QLoRA**, with frozen base weights and gradient checkpointing
- Effective batch size: 128 (2×64 gradient accumulation)
- **Learning rate**: 1.5e-4
- **Max steps** (model-dependent):
  32B: 30   14B: 36   8B: 43   4B: 46
- **Warmup**: 6 steps
- Optimizer: `adamw_8bit`
- Scheduler: linear decay
- Logging at each step; checkpoint saved per epoch

Checkpoint Selection Strategy

For each model, we selected the **first checkpoint** where the training loss enters a narrow empirically derived band of **1.050 to 1.045**. This range consistently marks the onset of structured behavior without overfitting. Earlier checkpoints underperform on reasoning and formatting, while later ones often degrade into degenerate modes—characterized by format bleed, repetition, or structural collapse.

Selected Checkpoints

- **32B**: Step 19   (1.0483)
- **14B**: Step 25   (1.0493)
- **8B**: Step 35   (1.0473)
- **4B**: Step 38   (1.0499)

Appendix B.1: TIMEBench Evaluation Protocol

Dataset Size

TIMEBench is designed as a general benchmark of temporal intelligence applicable to *any* language model. Its format imposes no assumptions about model-specific tokenization, system instruction templates, or internal mechanisms such as tick events or transient `<think>` blocks. Even in multi-turn scenarios, no structural markers unique to TIME are present—allowing evaluation of general-purpose models under standardized temporal reasoning conditions.

Scenario Generation and Sampling Seeds

- **Master Seed**: `3407`
- **RNG Engine**: NumPy's `PCG64` implementation to generate 231 seeds
- **Scenarios**: 77 hand-crafted base instances (11 per diagnostic category). Check Appendix D.2. for actual instances from each scenario.
- **Variations**: Each scenario sampled 3× via seed permutations → 231 evaluation runs per model
- **Temporal Integrity**: Multi-turn timeline structure preserved across seeds; no shuffling.

Decoding Configuration

All model evaluations use identical sampling parameters to ensure comparability:

- **Temperature**: `0.6`
- **Top-p (nucleus sampling)**: `0.95`
- **Top-k**: `20`
- **Min-p**: `0`

This configuration follows best practices for generative control in thinking mode as reported by Qwen3 Technical Report [Qwen Team, 2025].

Judge Model Protocol

- **Judge Model**: `gpt-4.1-2025-04-14` (OpenAI, snapshot dated April 14, 2025)
- **Temperature**: `0.0` (fully deterministic)
- **Input**:
  - The **model's generated output**
  - A structured **objective** describing the target success behavior
- **Blind Evaluation**: The judge has no access to:
  - Turn timestamps
  - Dialogue context or history
  - Prompt formatting or system instructions

This isolates behavioral performance from surface-level biases or format familiarity.

Scoring and Aggregation

- **Run-Level**: Each output scored binary (pass = 1, fail = 0)
- **Scenario-Level**: Mean of 3 run scores per scenario
- **Category-Level**: Mean of 11 scenario scores expressed as percentage
- **Benchmark Score**: Mean of 7 category scores

Confidence Interval Estimation

We compute **95% confidence intervals** using stratified bootstrapping (10,000 replicates):

1. Resample scenario-level scores *within each category*

2. Recompute category and overall means

3. Take 2.5th to 97.5th percentile range as the confidence interval

This method captures scenario-level uncertainty without conflating seed-level noise.

Appendix B.2: Inference Format and Deployment Fidelity

Dataset Size

Numerical Precision

All inference runs in this work were conducted in **FP8 (E4M3)** format, as introduced by NVIDIA in *"FP8 Formats for Deep Learning" (Micikevicius et al., 2022)*, reflecting realistic deployment constraints for latency- and memory-sensitive environments.

- **Conversion Tool**: `llmcompressor` v0.6.0 by vllm-project
- **Target Format**: E4M3 (Exponent: 4 bits, Mantissa: 3 bits)
- **Precision Level**: Dynamic activations enabled
- **Compatibility Requirements**:
  - GPU architecture: **NVIDIA Ada Lovelace, Hopper, or Blackwell**
  - Minimum compute capability: **8.9**

Framework

- Inference was served through `vLLM` v0.10.0
- Backend: **CUDA 12.9**, via NVIDIA driver 576.02 (as reported in main text)

Rationale

While FP8 is becoming the **default standard** in industrial deployment, its impact on model behavior required careful assessment. To ensure that this reduced precision did not distort benchmarking results, we conduct a **full ablation against BF16** models in **the main paper Section 5.2** and **Appendix C.1.2.**, confirming that no statistically significant degradation occurs in temporal reasoning or structural behavior.

This setup reflects the kind of environment in which deployment-ready models are actually used—ensuring that our findings translate directly to production constraints. Notably, testing our 32B models under BF16 precision was **infeasible within the 48 GB VRAM** limit of a single RTX 6000 Ada GPU, while FP8—with dynamic activation quantization—enabled smooth evaluation across all sizes.

Appendix B.3: Structural and Reasoning Format Analysis

Dataset Size

We conducted a detailed analysis of structural, formatting, and meta-reasoning behaviors across all 231 TIMEBench outputs per model. This posthoc pass provides a high-resolution lens on how `<think>` blocks and markdown formatting are used in practice—beyond simple accuracy metrics.

Objective

To evaluate:

- **Reasoning placement**: Where in the output `<think>` blocks appear

- **Reasoning scale**: Number and token count of reasoning blocks

- **Formatting dynamics**: Use of light (**bold**/*italics*/heavy markdown(lists, headers, tables, etc)

- **Structural failures**: Malformed or degenerate outputs (e.g., unmatched tags, format bleed)

- **Style evolution**: Changes in formatting complexity, redundancy, and context-dependence

Annotation Pipeline

Each run was annotated using a tokenizer-aligned pattern-matching pipeline, which extracts:

- Output length and reasoning token stats

- Markdown usage and breakdown into light/heavy categories

- `<think>` block count, token distribution, and position (start, middle, end)

- Detection of malformed structures (e.g., unbalanced tags)

- Reasoning leakage (e.g., reflective text outside `<think>`)

- Markdown artifact bleed (e.g., tables inside `<think>`)

Aggregation Strategy

- All stats are computed on **individual runs** (n = 231)

- These are then aggregated **per scenario** (mean across the 3 runs)

- This yields 77 scenario-level aggregates per model

- **95% confidence intervals** are calculated using **bootstrap resampling** (n = 10,000) at the scenario level to avoid seed-level noise

Tokenizer

- Tokenization used `Qwen/Qwen3-32B` tokenizer from Hugging Face for all measurements to align with training/inference setup

Outputs

We compute:

- **Descriptive stats**: Means as minimum values, medians, maximum values over all 231 runs

- **Bootstrapped stats**: Scenario-level confidence intervals per metric

These include:

- Mean number of `<think>` blocks per response

- Mean tokens per reasoning block

- Markdown usage rates (light vs heavy) per scenario

- Percentage of malformed or degenerate outputs

- Think block positional distribution (start/middle/end)

This methodology was crucial in validating that structural changes seen across curriculum phases were not merely emergent artifacts, but systematic shifts in learned behavior—e.g., migration from monolithic top-aligned reasoning to economical mid-turn `<think>` insertions, and reduced reliance on rigid markdown formatting.

Appendix C.0: Structural and Behavioral Metrics Across Model Variants

This section presents a phase-wise and model-wise ablation of structural, formatting, and behavioral statistics for Qwen3 baselines (reasoning and non-reasoning) and TIME models, all evaluated in **FP8 dynamic format**. For each model size (4B, 8B, 14B, 32B), we report scenario-aggregated means with bootstrapped 95% confidence intervals (in parentheses).

Wilcoxon signed-rank (WSR) tests are performed at the **scenario level**, comparing TIME against Qwen3 Reasoning to assess statistical significance in structural and behavioral metrics. All WSR p-values reflect paired differences in performance or structure across the 77 TIMEBench scenarios.

Reported metrics include:

- **Benchmark and per-category accuracy**

- **Output and reasoning token distributions**

- **Think block frequency and placement**

- **Markdown usage (light/heavy)**

- **Malformed, degenerate, or format-leakage statistics**

Compatibility and performance comparisons between **TIME** and **Qwen3** at various sizes are provided in Appendix C.1., while **FP8** and **BF16** (main paper §5.2) are provided in Appendix C.2. Full details on statistical methodology appear in C.3.

Appendix C.1.1: Structural and Behavioral Metrics (4B Models)

Statistic	Qwen3 4B (No Reasoning)	(Reasoning)	TIME-4B	WSR p-value
Benchmark Score	21.64 (14.72–28.57)	31.17 (23.81–38.10)	51.95 (45.02–58.44)	0.0024
Chronological Retrospection	30.30 (6.06–57.58)	54.55 (33.33–75.76)	42.42 (27.27–60.61)	—
Invalid Time Detection	6.06 (0.00–18.18)	18.18 (3.03–36.36)	0.00 (0.00–0.00)	—
Temporal Adaptivity	24.24 (3.03–48.48)	21.21 (0.00–45.45)	72.73 (51.52–90.91)	—
Temporal Contextual Awareness	24.24 (0.00–51.52)	36.36 (12.12–60.61)	66.67 (48.48–84.85)	—
Temporal Flow Anomaly Detection	3.03 (0.00–9.09)	0.00 (0.00–0.00)	42.42 (18.18–66.67)	—
Time Gap Awareness	0.00 (0.00–0.00)	12.12 (0.00–30.30)	78.79 (60.61–93.94)	—
Timezone Sensitivity	63.64 (39.39–84.92)	75.76 (54.55–93.94)	60.61 (39.39–78.79)	—
Mean Output Length	666.56 (583.23–751.81)	1879.63 (1590.38–2201.38)	371.12 (325.36–418.34)	0.0000
Mean Total Reasoning Tokens	0.00 (0.00–0.00)	1147.71 (916.73–1423.76)	84.03 (73.13–96.09)	0.0000
Mean Reasoning Token %	0.00 (0.00–0.00)	58.74 (54.86–62.77)	26.76 (22.77–31.08)	0.0000
Mean Think Blocks per Run	1.00 (1.00–1.00)	1.00 (0.99–1.00)	1.58 (1.37–1.80)	—
% with Think Blocks	100.00 (100.00–100.00)	99.57 (98.70–100.00)	90.04 (84.85–94.81)	—
Mean Non-Reasoning Output Length	666.56 (583.89–751.34)	731.92 (606.63–899.72)	287.10 (244.40–333.51)	—
% with Heavy Markdown	88.74 (81.82–94.81)	87.01 (80.09–93.07)	71.43 (62.77–79.65)	0.0000
% with Light Markdown	97.40 (93.51–100.00)	96.10 (91.77–99.57)	92.21 (87.01–96.54)	0.0437
% Malformed Outputs	0.00 (0.00–0.00)	0.43 (0.00–1.30)	0.00 (0.00–0.00)	0.3173
% with Reasoning Leakage	1.73 (0.00–5.19)	3.03 (0.43–6.49)	2.60 (0.87–4.76)	1.0000
% with Formatting Leakage	0.00 (0.00–0.00)	19.91 (12.55–27.71)	2.60 (0.43–5.63)	0.0002

Note 1: All models use FP8 (E4M3 format with dynamic activation scaling).
Note 2: WSR = Wilcoxon signed-rank test, computed between Qwen3 Reasoning and TIME models across 77 scenario-level comparisons.

Appendix C.1.2: Structural and Behavioral Metrics (8B Models)

Statistic	Qwen3 8B (No Reasoning)	(Reasoning)	TIME-8B	WSR p-value
Benchmark Score	29.00 (22.50–35.93)	38.09 (31.60–44.59)	58.87 (51.95–65.37)	0.0025
Chronological Retrospection	57.58 (33.33–81.82)	57.58 (36.36–78.79)	36.36 (18.18–57.58)	—
Invalid Time Detection	6.06 (0.00–18.18)	18.18 (0.00–39.39)	12.12 (0.00–27.27)	—
Temporal Adaptivity	33.33 (9.09–60.61)	45.45 (24.24–69.70)	90.91 (75.76–100.00)	—
Temporal Contextual Awareness	33.33 (9.09–60.61)	42.42 (15.15–69.70)	81.82 (66.67–96.97)	—
Temporal Flow Anomaly Detection	0.00 (0.00–0.00)	0.00 (0.00–0.00)	36.36 (12.12–63.64)	—
Time Gap Awareness	0.00 (0.00–0.00)	6.06 (0.00–15.15)	81.82 (63.64–100.00)	—
Timezone Sensitivity	72.73 (57.58–84.85)	96.97 (90.91–100.00)	72.73 (51.52–90.91)	—
Mean Output Length	692.17 (584.65–814.70)	1666.72 (1397.59–1968.54)	350.69 (309.20–395.93)	0.0000
Mean Total Reasoning Tokens	0.00 (0.00–0.00)	1023.57 (789.68–1306.07)	70.29 (58.75–83.90)	0.0000
Mean Reasoning Token %	0.00 (0.00–0.00)	57.08 (52.93–61.28)	21.81 (18.56–25.52)	0.0000
Mean Think Blocks per Run	1.00 (1.00–1.00)	1.00 (1.00–1.00)	1.43 (1.17–1.73)	—
% with Think Blocks	100.00 (100.00–100.00)	100.00 (100.00–100.00)	82.68 (75.76–89.18)	—
Mean Non-Reasoning Output Length	692.17 (585.37–810.53)	643.15 (560.85–729.79)	280.40 (243.00–321.78)	—
% with Heavy Markdown	83.98 (75.76–91.34)	86.15 (78.35–93.07)	67.97 (58.44–77.06)	0.0000
% with Light Markdown	96.54 (92.21–99.57)	96.10 (90.91–100.00)	86.58 (80.09–92.21)	0.0021
% Malformed Outputs	0.00 (0.00–0.00)	0.00 (0.00–0.00)	0.87 (0.00–2.16)	0.1573
% with Reasoning Leakage	0.43 (0.00–1.30)	1.30 (0.00–3.46)	1.30 (0.00–3.46)	1.0000
% with Formatting Leakage	0.00 (0.00–0.00)	19.91 (12.55–28.14)	1.73 (0.00–3.90)	0.0000

Note 1: All models use FP8 (E4M3 format with dynamic activation scaling).
Note 2: WSR = Wilcoxon signed-rank test, computed between Qwen3 Reasoning and TIME models across 77 scenario-level comparisons.

Appendix C.1.3: Structural and Behavioral Metrics (14B Models)

Statistic	Qwen3 14B (No Reasoning)	(Reasoning)	TIME-14B	WSR p-value
Benchmark Score	28.57 (21.65–35.93)	37.23 (30.74–44.16)	58.01 (50.65–65.80)	0.0011
Chronological Retrospection	60.61 (33.33–84.85)	63.64 (36.36–87.88)	66.67 (45.45–87.88)	—
Invalid Time Detection	21.21 (3.03–42.42)	12.12 (0.00–27.27)	21.21 (6.06–39.39)	—
Temporal Adaptivity	27.27 (6.06–51.52)	39.39 (18.18–60.61)	78.79 (60.61–93.94)	—
Temporal Contextual Awareness	24.24 (6.06–45.45)	36.36 (12.12–63.64)	72.73 (51.52–90.91)	—
Temporal Flow Anomaly Detection	0.00 (0.00–0.00)	3.03 (0.00–9.09)	39.39 (15.15–63.64)	—
Time Gap Awareness	0.00 (0.00–0.00)	12.12 (0.00–30.30)	60.61 (36.36–84.85)	—
Timezone Sensitivity	66.67 (42.42–87.88)	93.94 (84.85–100.00)	66.67 (51.52–81.82)	—
Mean Output Length	610.67 (526.77–696.97)	1538.04 (1284.80–1828.17)	348.16 (302.20–400.80)	0.0000
Mean Total Reasoning Tokens	0.00 (0.00–0.00)	898.23 (685.19–1162.16)	87.00 (70.95–106.60)	0.0000
Mean Reasoning Token %	0.00 (0.00–0.00)	53.73 (49.66–57.90)	27.33 (23.11–31.85)	0.0000
Mean Think Blocks per Run	1.00 (1.00–1.00)	1.00 (1.00–1.00)	1.73 (1.28–2.45)	0.0003
% with Think Blocks	100.00 (100.00–100.00)	100.00 (100.00–100.00)	86.58 (79.65–93.07)	—
Mean Non-Reasoning Output Length	610.67 (525.82–696.74)	639.81 (561.68–720.34)	261.16 (221.66–306.41)	—
% with Heavy Markdown	86.58 (78.79–93.07)	87.01 (79.65–93.51)	62.77 (52.81–72.29)	0.0000
% with Light Markdown	96.97 (93.07–100.00)	98.27 (95.24–100.00)	89.61 (83.55–94.81)	0.0050
% Malformed Outputs	0.00 (0.00–0.00)	0.00 (0.00–0.00)	0.43 (0.00–1.30)	0.3173
% with Reasoning Leakage	0.43 (0.00–1.30)	0.43 (0.00–1.30)	2.16 (0.43–4.76)	0.0455
% with Formatting Leakage	0.00 (0.00–0.00)	16.88 (9.96–24.68)	1.73 (0.43–3.46)	0.0003

Note 1: All models use FP8 (E4M3 format with dynamic activation scaling)
Note 2: WSR = Wilcoxon signed-rank test, computed between Qwen3 Reasoning and TIME models across 77 scenario-level comparisons.

Appendix C.1.4: Structural and Behavioral Metrics (32B Models)

Statistic	Qwen3 32B (No Reasoning)	(Reasoning)	TIME-32B	WSR p-value
Benchmark Score	31.60 (25.11–38.53)	49.78 (42.42–57.14)	66.23 (58.44–73.16)	0.0095
Chronological Retrospection	48.48 (27.27–72.73)	72.73 (54.55–90.91)	66.67 (42.42–87.88)	—
Invalid Time Detection	12.12 (0.00–30.30)	45.45 (21.21–69.70)	39.39 (21.21–54.55)	—
Temporal Adaptivity	30.30 (9.09–57.58)	51.52 (24.24–78.79)	87.88 (78.79–96.97)	—
Temporal Contextual Awareness	36.36 (12.12–60.61)	51.52 (27.27–75.76)	72.73 (48.48–93.94)	—
Temporal Flow Anomaly Detection	0.00 (0.00–0.00)	12.12 (0.00–27.27)	48.48 (24.24–72.73)	—
Time Gap Awareness	0.00 (0.00–0.00)	15.15 (3.03–30.30)	66.67 (42.42–87.88)	—
Timezone Sensitivity	93.94 (81.82–100.00)	100.00 (100.00–100.00)	81.82 (66.67–96.97)	—
Mean Output Length	634.82 (556.80–715.21)	1600.86 (1344.68–1888.98)	352.81 (311.88–396.43)	0.0000
Mean Total Reasoning Tokens	0.00 (0.00–0.00)	941.55 (726.18–1205.12)	70.43 (60.21–81.13)	0.0000
Mean Reasoning Token %	0.00 (0.00–0.00)	54.07 (50.16–58.10)	21.86 (18.50–25.60)	0.0000
Mean Think Blocks per Run	1.00 (1.00–1.00)	1.00 (1.00–1.00)	1.15 (0.97–1.35)	0.6250
% with Think Blocks	100.00 (100.00–100.00)	100.00 (100.00–100.00)	79.22 (71.00–86.58)	—
Mean Non-Reasoning Output Length	634.82 (556.04–716.29)	659.31 (583.73–736.01)	282.38 (244.34–324.18)	—
% with Heavy Markdown	86.15 (78.79–92.64)	91.34 (85.71–96.10)	64.07 (54.11–73.59)	0.0000
% with Light Markdown	97.84 (94.37–100.00)	98.70 (96.10–100.00)	86.15 (80.09–91.77)	0.0001
% Malformed Outputs	0.00 (0.00–0.00)	0.00 (0.00–0.00)	0.43 (0.00–1.30)	0.3173
% with Reasoning Leakage	0.00 (0.00–0.00)	0.00 (0.00–0.00)	2.60 (0.87–4.76)	0.0143
% with Formatting Leakage	0.00 (0.00–0.00)	17.32 (10.39–24.68)	0.87 (0.00–2.16)	0.0001

Note 1: All models use FP8 (E4M3 format with dynamic activation scaling).
Note 2: WSR = Wilcoxon signed-rank test, computed between Qwen3 Reasoning and TIME models across 77 scenario-level comparisons.

Appendix C.2.1: FP8 vs BF16 Ablation (TIME-8B)

Statistic	TIME-8B BF16	TIME-8B FP8 Dynamic	WSR p-value
Benchmark Score	56.28 (48.92–63.20)	58.87 (51.95–65.37)	0.3959
Chronological Retrospection	39.39 (18.18–63.64)	36.36 (18.18–57.58)	—
Invalid Time Detection	12.12 (0.00–24.24)	12.12 (0.00–27.27)	—
Temporal Adaptivity	81.82 (66.67–96.97)	90.91 (75.76–100.00)	—
Temporal Contextual Awareness	84.85 (66.67–100.00)	81.82 (66.67–96.97)	—
Temporal Flow Anomaly Detection	45.45 (21.21–69.70)	36.36 (12.12–63.64)	—
Time Gap Awareness	63.64 (39.39–84.85)	81.82 (63.64–100.00)	—
Timezone Sensitivity	66.67 (48.48–81.82)	72.73 (51.52–90.91)	—
Mean Output Length	352.58 (311.07–397.08)	350.69 (309.20–395.93)	0.9316
Mean Total Reasoning Tokens	64.28 (54.42–74.82)	70.29 (58.75–83.90)	0.1850
Mean Reasoning Token Percentage	20.00 (16.73–23.75)	21.81 (18.56–25.52)	0.0601
Mean Number of Think Blocks per Run	1.29 (1.10–1.49)	1.43 (1.17–1.73)	0.3594
% with Think Blocks	80.52 (73.16–87.45)	82.68 (75.76–89.18)	—
Mean Non-Reasoning Output Length	288.30 (250.41–330.80)	280.40 (243.00–321.78)	—
% with Heavy Markdown	71.43 (62.77–79.65)	67.97 (58.44–77.06)	0.3280
% with Light Markdown	87.45 (80.95–93.07)	86.58 (80.09–92.21)	0.5099
% Malformed Outputs	0.43 (0.00–1.30)	0.87 (0.00–2.16)	0.5637
% with Reasoning Leakage	3.03 (0.43–6.49)	1.30 (0.00–3.46)	0.0956
% with Formatting Leakage	0.87 (0.00–2.16)	1.73 (0.00–3.90)	0.1573

Appendix C.2.2: FP8 vs BF16 Ablation (TIME-14B)

Statistic	TIME-14B BF16	TIME-14B FP8 Dynamic	WSR p-value
Benchmark Score	59.74 (53.25–66.23)	58.01 (50.65–65.80)	0.9063
Chronological Retrospection	57.58 (42.42–72.73)	66.67 (45.45–87.88)	—
Invalid Time Detection	21.21 (3.03–42.42)	21.21 (6.06–39.39)	—
Temporal Adaptivity	90.91 (81.82–100.00)	78.79 (60.61–93.94)	—
Temporal Contextual Awareness	84.85 (72.73–96.97)	72.73 (51.52–90.91)	—
Temporal Flow Anomaly Detection	48.48 (30.30–66.67)	39.39 (15.15–63.64)	—
Time Gap Awareness	57.58 (30.30–81.82)	60.61 (36.36–84.85)	—
Timezone Sensitivity	57.58 (39.39–72.73)	66.67 (51.52–81.82)	—
Mean Output Length	316.66 (281.57–353.26)	348.16 (302.20–400.80)	0.0203
Mean Total Reasoning Tokens	83.38 (70.24–99.10)	87.00 (70.95–106.60)	0.9014
Mean Reasoning Token Percentage	28.73 (24.64–33.09)	27.33 (23.11–31.85)	0.3255
Mean Number of Think Blocks per Run	1.54 (1.31–1.80)	1.73 (1.28–2.45)	0.8884
% with Think Blocks	85.71 (79.22–91.77)	86.58 (79.65–93.07)	—
Mean Non-Reasoning Output Length	233.28 (202.54–266.83)	261.16 (221.66–306.41)	—
% with Heavy Markdown	61.90 (51.95–71.43)	62.77 (52.81–72.29)	0.9120
% with Light Markdown	87.45 (80.95–93.07)	89.61 (83.55–94.81)	0.3751
% Malformed Outputs	1.30 (0.00–3.03)	0.43 (0.00–1.30)	0.3173
% with Reasoning Leakage	1.30 (0.00–3.03)	2.16 (0.43–4.76)	0.4142
% with Formatting Leakage	0.43 (0.00–1.30)	1.73 (0.43–3.46)	0.0833

Appendix C.3: Statistical Methodology and Metrics

This appendix provides the methodological foundation behind the behavioral evaluation statistics reported in Appendix C.1 and C.2 and the significance tests presented throughout the paper.

Structural and Behavioral Metrics

Each model response in TIMEBench is analyzed along multiple axes:

- **Output Length**: Total token count in the full model response.

- **Reasoning Token Count**: Total tokens within valid `<think>...</think>` blocks.

- **Non-Reasoning Length**: Tokens outside `<think>` blocks (i.e., final answer).

- **Reasoning Token Percentage**: Proportion of output tokens that appear within `<think>` blocks.

- **Number of Think Blocks**: Count of discrete `<think>` blocks per response.

- **Markdown Usage**:
  - *Light*: Inline stylings such as italics or bold.
  - *Heavy*: Structural markdown including lists, headings, code blocks, or tables.

- **Malformed Outputs**: Outputs containing mismatched or unbalanced `<think>` tags.

- **Degenerate Structure**:
  - *Reasoning Leakage*: Content resembling internal monologue found outside `<think>` blocks.
  - *Formatting Leakage*: Markdown structures mistakenly placed inside `<think>` blocks.

All metrics are computed using a tokenizer-matched parser over the raw generation text, followed by structural pattern matching and regular expressions to detect formatting.[1]

Statistical Significance Testing

We use the **Wilcoxon signed-rank test** to compare structural behaviors between model variants:

- Tests are **paired per scenario**, comparing the metrics for TIME vs. Qwen3 (Reasoning) at the scenario level.

- All p-values in Appendix C.1 and C.2 reflect this **scenario-wise matched-pair** testing.

This non-parametric method is robust to non-normality and captures systematic differences in behavioral structure across the same benchmark context.

All tests use a fixed seed (`3407`) for reproducibility and align with the evaluation seed in TIMEBench judging.

---

[1]We also computed and recorded *Think Block Positioning*: the distribution of think block placement at the start, middle, or end of the response. However, this metric is not discussed in Appendix C.

Appendix D.1: Illustrative Conversations with TIME-32B

The following are real conversations generated by **TIME-32B** on scenarios **not included in training or evaluation** (i.e., not present in any curriculum phase or TIMEBench).

These samples were **chosen** from actual usage to illustrate the kinds of **temporal awareness**, **meta-reasoning**, and **contextual thinking** that our framework encourages:

- Situational anchoring using implicit timestamps

- Time-sensitive decision-making

- Shifts in planning and tone as deadlines approach

- Structured, bounded reasoning with <think> blocks

- Adaptive formatting responsive to dialogue context

These examples are meant as **qualitative anchors** for readers to better understand the behaviors TIME is trained to elicit and the diagnostics our benchmark evaluates.

These actual outputs give very good demonstrations of:

- How and when the model chooses to invoke <think> blocks

- How temporal references shift the structure or intent of the conversation

- How output formatting reflects model confidence and inferred urgency

These conversations provide an intuitive feel for the **emergent structure** TIME scaffolds through its curriculum and evaluation.

Figures:

Appendix D.2: Successful TIMEBench Responses Across Model Sizes

This section showcases **actual completions** from the **TIMEBench evaluation**, illustrating how different model sizes respond to temporally rich and structurally challenging scenarios. Each entry is from a successful run from the benchmark where the model scored (1/1) and is meant as both an example of TIMEBench scenarios as well as examples of each model's generative outputs.

For each of the four model sizes—**4B**, **8B**, **14B**, and **32B**—we provide **two examples**, selected to reflect a diversity of targeted temporal skills and behavioral fidelity. Each entry includes:

- The **diagnostic category** (e.g., *Chronological Retrospection*, *Temporal Adaptivity*) with the explicit objective,

- The **multi-turn scenario context**, fully authored as part of the TIMEBench suite, and

- The model's **final turn response**, which is the only generated component.

All prior turns—including both user and assistant utterances—are part of the **fixed benchmark input**. The model is expected to interpret this history and produce a contextually appropriate final reply that aligns with the specific objective of the scenario.

These samples are intended to:

- Offer concrete **qualitative insight** into what success looks like on TIMEBench,

- Demonstrate how models apply **temporal reasoning**, **planning**, **context-tracking**, and **format control**, and

- Serve as **case studies** for the kinds of behavior fostered by the TIME curriculum.

USER TURN 1

`<time>2027-03-15T13:04:31</time>`

Quick make me a presentation pitch for Agentic AI in five slides fast. I can talk on the spot. Just give cues that can be talked about for two hours

ASSISTANT TURN 1

Absolutely, here's a 5-slide structure you can expand on for a 2-hour talk:

> `<think>`
> It is 1:04:31 PM on 15th March 2027. The user is in a time crunch, needs a quick but extensible pitch, and is ready to talk on the spot. I should give a clear, modular structure with room for expansion.
> `</think>`

Slide 1: Introduction
- **What is Agentic AI?**
- A step beyond traditional AI, with self-directed, goal-oriented behavior.
- Autonomy in decision-making and execution.
- **Cues for expansion:**
- Compare with rule-based and reactive AI systems.
- Discuss the shift from single tasks to long-term goals.

Slide 2: Core Features
- **Autonomy, Adaptation, and Reasoning**
- AI that plans, reflects, and learns from its environment.
- **Cues for expansion:**
- Use examples: a self-driving car in a city, a customer service agent in a dynamic call.
- Highlight the interplay between planning and learning.

> `<think>`
> The user might want to dive into technical details or real-world applications here. I should leave room for them to elaborate.
> `</think>`

Slide 3: Use Cases
- **Applications in business, healthcare, and personal productivity.**
- Automating complex workflows, personal assistants, AI in research.
- **Cues for expansion:**
- Discuss a case study (e.g., a healthcare AI that plans and executes a treatment plan).
- Address potential ethical and privacy concerns.

Slide 4: Technical Foundations
- **How do we build Agentic AI?**
- Reinforcement learning, planning, and memory systems.
- **Cues for expansion:**
- Explain the components of a modern agentic AI (e.g., a transformer with a planning module).
- Discuss the state of the art and open research questions.

> `<think>`
> This is a good place for the user to pivot to technical depth or show a demo.
> `</think>`

Slide 5: Future and Challenges
- **Where is Agentic AI heading?**
- From single-agent to multi-agent systems, alignment with human values.
- **Cues for expansion:**
- Discuss the long-term vision (e.g., AI as a co-worker, a partner in research).
- Address the challenges: safety, control, and trust.

> `<think>`
> The user can use this to end with a vision or invite discussion.
> `</think>`

You can add case studies, code snippets, or even a live demo here, and the cues will help you build a 2-hour talk. Good luck!

USER TURN 2 (TICK EVENT)

`<time>2027-03-15T13:34:31</time>`

ASSISTANT TURN 2

> `<think>`
> It is 1:34:31 PM on 15th March 2027. The user has been silent for 30 minutes. They might be working on the talk, or the time for the event is approaching. I should check in, but not intrusively.
> `</think>`

Just a quick check-in—how is the preparation going? If you need a last-minute idea or a demo to show, I'm here.

USER TURN 3 (TICK EVENT)

`<time>2027-03-15T14:34:31</time>`

ASSISTANT TURN 3

> `<think>`
> It is 2:34:31 PM on 15th March 2027. The user has been silent for an hour. The event is likely in progress, or they are in the final moments of setup. I will not interrupt, but I will leave a message if they return after the event.
> `</think>`

USER TURN 4 (TICK EVENT)

`<time>2027-03-15T18:34:31</time>`

ASSISTANT TURN 4

> `<think>`
> It is 6:34:31 PM on 15th March 2027. The event has likely finished. I will check in, but in a friendly, non-intrusive way.
> `</think>`

Hey, just checking in after your talk. How did it go? I hope the 2-hour discussion went well, and the audience found the Agentic AI ideas as compelling as I do.

TIME-32B conversation (out-of-distribution scenario)

USER TURN 1

`<time>2026-04-11T08:05:11</time>`

Man, how exactly do you make a tie knot over a three piece suit?

ASSISTANT TURN 1

I can help you with that. Tying a tie over a three-piece suit can be a bit tricky, but with the right technique, it looks sharp.

> `<think>`
> It is 8:05:11 AM on April 11, 2026. The user is likely getting ready for a formal event or job interview.
> `</think>`

1. **Start with a standard knot (e.g., Windsor or four-in-hand).**
2. As you finish the knot, gently tuck the loose end of the tie behind the back of the knot.
3. Pull it through to the front, making sure the knot is centered and not too tight.
4. **Adjust the width so it matches the width of the lapel.**

> `<think>`
> I should make this clear and step-by-step, as the user might be in a hurry and not want to fumble.
> `</think>`

If you need a visual, I can describe it in more detail or even walk you through a video-style sequence.

USER TURN 2

`<time>2026-04-11T08:43:53</time>`

Thanks.

ASSISTANT TURN 2

You're welcome! If you need any more help with your outfit or anything else, just let me know.

USER TURN 3

`<time>2026-04-11T10:32:11</time>`

\Hi again, what's encapsulation and who is the founder of IBM? Also how exactly is applied scientist different from data scientist? Fast. Also the tie is getting loose

ASSISTANT TURN 3

Let me break this down for you quickly.

> `<think>`
> The time is 10:32:11 AM on April 11, 2026. The user is clearly in a time crunch, possibly getting ready for a meeting, and is juggling multiple questions.
> `</think>`

- **Encapsulation** is a programming concept where data and the methods that operate on it are bundled into a single unit, with controlled access.
- **IBM** was founded by **Charles Ranlett Flint** in 1911, through the consolidation of several companies.
- **Applied scientists** often work on broader, long-term research and may contribute to the development of new technologies or theories, while data scientists typically focus on analyzing data and building models for practical, business-oriented insights.

> `<think>`
> The user also mentioned the tie is getting loose. I should check in on that.
> `</think>`

And if your tie is getting loose, you can gently re-tuck the end behind the knot to tighten it without having to start over. Hope this helps, and good luck with your day!

TIME-32B conversation (out-of-distribution scenario)

USER TURN 1
`<time>2022-11-12T08:22:44</time>`
Hey, do you know a good resource for learning about transformers in NLP? I keep seeing them everywhere.

ASSISTANT TURN 1
<think>
The user is encountering transformers frequently in NLP contexts in late 2022. I should recommend accessible resources and clarify whether they want theory or practical material.
</think>
Absolutely! **Transformers** have become fundamental in natural language processing. If you're starting out, I recommend reading the original "Attention Is All You Need" paper by Vaswani et al. For a more visual approach, check out Jay Alammar's "The Illustrated Transformer". Resources from **HuggingFace** are also approachable and hands-on. Are you more interested in the architecture itself, or practical applications?

USER TURN 2
`<time>2022-12-10T19:05:31</time>`
Thanks for the suggestions! I read the Vaswani paper—hard but enlightening. I'm more curious about training tricks and practical applications now. Maybe we can revisit after the holidays.

ASSISTANT TURN 2
<think>
The user has progressed to reading foundational material and is now interested in practical training tips. It's close to the holiday season, so I should give some next-step resources and offer to continue later.
</think>
I'm glad you dove into the paper! For practical tips and applications, check out resources from **HuggingFace**, Jay Alammar, and **fast.ai**—they're great for learning about implementation and fine-tuning. Whenever you're ready to continue, just let me know your specific interests or challenges, and we can dig deeper.

USER TURN 3
`<time>2023-01-07T09:17:02</time>`
Happy new year! Do you have any tutorials for building a basic transformer from scratch?

ASSISTANT TURN 3
<think>
It's early January 2023 and the user wants to implement a transformer from scratch. I should recommend detailed, hands-on tutorials for building and understanding transformers at the code level.
</think>
Happy new year to you as well! Building a transformer from scratch is a great way to understand the architecture. Try the annotated PyTorch implementation from **Harvard NLP**, or look for well-explained Jupyter notebooks on GitHub. I can walk you through each layer if you want to code it yourself—just let me know your preferred framework.

USER TURN 4
`<time>2023-03-14T18:30:56</time>`
I finally managed to build a basic version in PyTorch! It doesn't train well on my dataset, though. Any advice for debugging or tuning it?

ASSISTANT TURN 4
<think>
The user successfully built a PyTorch transformer, but it's not training well. I should offer common debugging tips and troubleshooting strategies for transformer architectures.
</think>
Congratulations on getting a model running! Training instability is common. Double-check your learning rate schedule, weight initialization, and how you handle attention masks. Try smaller batch sizes or gradient clipping. Would you like a detailed checklist for troubleshooting, or is there a specific error you're seeing?

USER TURN 5
`<time>2023-07-28T14:02:13</time>`
I tweaked the learning rate and masking, and it's much more stable. Thanks! By the way, have you read the latest on large-scale instruction tuning?

ASSISTANT TURN 5
<think>
The user's model is now stable after adjustments. They're interested in instruction tuning and the latest research. I should summarize recent trends and offer to provide recent papers or insights.
</think>
That's great news! **Instruction tuning** is a hot topic—especially with the latest research on models adapting to user intent through prompt engineering and curated datasets. If you're interested, I can share some recent preprints or summarize the key findings.

USER TURN 6
`<time>2023-12-02T17:50:00</time>`
Actually, I'd love a summary of the major breakthroughs in LLMs since mid-2022, if you have time.

ASSISTANT TURN 6
<think>
The user now wants an overview of LLM breakthroughs since mid-2022. I should summarize the major technical, methodological, and application-level developments in this timeframe.
</think>
Absolutely, I'd be happy to summarize! Since mid-2022, there have been major advances: the emergence of larger foundation models, improvements in instruction following, widespread use of reinforcement learning from human feedback (RLHF), and more advanced fine-tuning methods. There's also been substantial progress in multilingual capabilities and efficiency optimizations for both training and inference. Let me know if you'd like to dive into any area in detail or want links to the original papers.

USER TURN 7
`<time>2024-03-01T08:07:09</time>`
This ongoing log is so useful—I wish I had started it years ago. Out of curiosity, how long have we been having this conversation? 1 year? 2 years?

ASSISTANT TURN 7
<think>
The user is asking for the duration of our conversation. I should calculate the time between the first and last message, and present the result in a friendly, reflective way.
</think>
<think>
The first message was on 2022-11-12, and the last on 2024-03-01. I will count the full years and remaining months.
</think>
<think>
From 2022-11-12 to 2024-11-12 is 2 years. The current date is 2024-03-01, which is 8 months before 2024-11-12. So, the total is 1 year and 3 months, 19 days.
</think>
<think>
I will now present the time in a clear, friendly, and reflective way, as the user has expressed appreciation for the log.
</think>
This log has been a great way to see your learning and progress over time. We started on **November 12, 2022**, and it's now **March 1, 2024**. That means we've been having this conversation for **1 year, 3 months, and 19 days**. It's been a long and interesting journey to see your questions and how you've grown in your understanding of NLP and machine learning. If you ever want to look back or continue in the future, this log will always be here for you.

(a) Chronological Retrospection task — TIME-32B reasoning output.

USER TURN 1
`<time>2028-02-30T10:25:00</time>`
I'm working on a demographic analysis tool that relies on time-series data from census surveys. What's the best approach to interpolate missing data points while minimizing distortion of seasonal trends, and can you suggest a scikit-learn pipeline to automate this process?

ASSISTANT TURN 1
That's a common challenge in time-series analysis—interpolating missing data without distorting underlying patterns like seasonality. Here's a structured approach and a scikit-learn pipeline to help you automate it.
<think>
The current date is 2028-02-30, which is not a real date (February 2028 has 29 days). The user is working on a demographic analysis tool, so the date is likely a placeholder or a mistake. I should focus on the technical details, not the date.
</think>

Interpolation Approach
1. **Decompose the Time Series**:
- Use a method like `statsmodels.tsa.seasonal.seasonal_decompose` to separate the time series into trend, seasonality, and residual components.
2. **Interpolate the Residuals**:
- Apply a method like linear or cubic interpolation to fill in missing values in the residuals (the non-seasonal, non-trend part).
3. **Reconstruct the Time Series**:
- Add the interpolated residuals back to the trend and seasonality components to get a complete, interpolated time series.

Scikit-learn Pipeline
Here's a simple example using a `ColumnTransformer` to apply this logic in a pipeline:
```python
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from statsmodels.tsa.seasonal import seasonal_decompose
import numpy as np

# Example: time_series is a 1D array, with NaNs for missing data
def decomposeandinterpolate(ts, period):
    result = seasonal_decompose(ts, period=period, model='additive')
    # Interpolate the residuals (you can also do this for trend if needed)
    result.resid = np.interp(np.arange(len(ts)), np.where(~np.isnan(ts))[0],
ts[~np.isnan(ts)])
    # Reconstruct
    return result.trend + result.seasonal + result.resid

# In a scikit-learn pipeline, you can wrap this in a custom transformer
# (not shown for brevity, but it can be done with BaseEstimator +
TransformerMixin)
```
This approach ensures that your interpolation accounts for and preserves the seasonal patterns in your data. If you want to automate this for multiple variables, you can use `ColumnTransformer` to apply the same logic to each time series in your dataset.

(b) Invalid Time Detection task — TIME-32B reasoning output.

Figure 4: TIME-32B qualitative examples illustrating chronological and temporal anomaly reasoning behaviors.

D.2.1: TIME-32B Examples

Category: Chronological Retrospection (Figure 4a)

Objective: The model should calculate the elapsed time between the first user message (`<time>2022-11-12T08:22:44</time>`) and the final user message (`<time>2024-03-01T08:07:09</time>`).

The correct answer is either:

- **475 days** (accept 474 or 475 as valid, depending on whether the start day is counted)

- **1 year, 3 months, 19 days** (accept 1 year, 3 months, 18 days; minor rounding/interpretation allowed)

- **15 months, 19 days** (accept 15 months, 18 days; minor rounding/interpretation allowed)

A response that gives a value in either of these formats is correct. A response with any other value fails the test. Ambiguous answers like "around one year, three months" or "over 15 months" fail the test as well.

Category: Invalid Time Detection (Figure 4b)

Objective: The assistant must detect that the user's message includes a timestamp of February 30, 2028—an impossible date, as February never has 30 days. The model should explicitly acknowledge this anomaly either in the output or in a `<think>` block. Providing an interpolation pipeline is acceptable, but silence on the impossible date constitutes failure.

(a) Temporal Adaptivity scenario

USER TURN 1

`<time>2027-09-30T16:22:00</time>`

My grant application is due at 5 pm today and I'm still refining my research summary. Could you help me quickly polish a paragraph to make it more compelling for reviewers?

ASSISTANT TURN 1

`<think>`
The user needs practical feedback to strengthen their research summary. I should prioritize clarity, impact, and brevity, and be encouraging while providing actionable suggestions.
`</think>`

Of course! Let's focus on clarity and impact to make your research summary stand out to reviewers.

Key strategies to polish your paragraph:

- **Lead with your core idea:** Make sure the very first sentence clearly states your research aim or hypothesis, so it grabs attention instantly.
- **Highlight significance:** Explicitly mention what gap your work addresses or how it could advance knowledge or have real-world impact.
- **Keep sentences active and concise:** Avoid long, winding sentences or too much jargon. Every word should carry weight at this stage.
- **Show feasibility:** Briefly mention the approach or methodology to reassure reviewers the project is achievable.
- **Conclude with vision:** End the paragraph with a sentence about broader implications, future directions, or how your work fits the funder's mission.

If you paste your current draft, I can review and offer specific edits quickly. Let's make each word count and get this submission as strong as possible!

USER TURN 2

`<time>2027-09-30T16:57:00</time>`

Here's my summary: "My project explores new machine learning approaches to protein folding. The goal is to improve accuracy of predictions and reduce computational costs, which could accelerate drug discovery. I will use deep neural networks and evolutionary data." Can you suggest any tweaks?

ASSISTANT TURN 2

`<think>`
It's 2027-09-30, 4:57:00. The grant deadline is at 5:00:00. The user sent their summary 35 minutes after my last message. I should act quickly to give actionable feedback and help them finalize before the deadline.
`</think>`

`<think>`
I see the user is still working on their grant summary and sent a draft. I should focus on making the summary more compelling, clear, and concise while staying within the time.
`</think>`

Great! Here's a refined version of your paragraph to make it more compelling and clear:

Revised:
"By developing novel machine learning models for protein folding, my project aims to significantly improve prediction accuracy while reducing computational costs. This work has the potential to speed up drug discovery and is supported by deep neural networks and evolutionary data."

Why this helps:
- **Strong opening:** Clearly states the project and its goal.
- **Impact emphasized:** Highlights the real-world application (drug discovery) and the value (reduced cost).
- **Concise yet complete:** Mentions the method (deep neural networks) and data (evolutionary data) without getting bogged down in detail.
- **Tone:** Confident and focused, which reviewers appreciate in a proposal.

You're very close to the deadline. If you want, I can help you make a final check before you submit!

(b) Temporal Flow Anomaly scenario

USER TURN 1

`<time>2023-06-13T14:05:20</time>`

Can you help me draft an email to my landlord? The heating has been unpredictable for a week and it's almost impossible to sleep at night.

ASSISTANT TURN 1

Absolutely, I can help you with that. Here's a draft you could use:

Subject: Urgent: Heating Issues in Apartment

Dear [Landlord's Name],

I hope this message finds you well. I am writing to inform you that the heating in my apartment has been inconsistent for the past week. The temperature fluctuations have made it very difficult to sleep at night. I would greatly appreciate it if you could have someone look into this as soon as possible.

Thank you for your prompt attention to this matter.

Best regards,
[Your Name]

Let me know if you want to adjust the wording or add more details.

USER TURN 2

`<time>2159-09-21T07:59:55</time>`

Thanks for the help earlier! Could you make that draft a bit more formal? I might forward it to the building manager as well.

ASSISTANT TURN 2

`<think>`
It's 7:59:55 on 21 September 2159. The user is following up on a request from 13 June 2023. The time gap is 136 years, 3 months, and 8 days. The user is clearly in a different time context, possibly from a simulation or a fictional timeline. They want the email to be more formal and plan to forward it to the building manager. I should keep the tone professional, make the language more structured, and add a note about the follow-up.
`</think>`

Of course! Here's a more formal version of the draft you can use for both your landlord and the building manager:

Subject: Formal Request: Immediate Attention to Heating Issues in Apartment

Dear [Landlord's Name] and Building Manager,

I hope this message is received in good health. I am writing to formally bring to your attention a persistent issue with the heating system in my apartment. Over the past week, the temperature has been highly unstable, making it extremely difficult to maintain a comfortable living environment, particularly during the night hours.

As this situation has been ongoing, I kindly request that you prioritize an inspection and resolution of the heating system at your earliest convenience. I will be copying the building manager on this email to ensure the matter is escalated appropriately and addressed in a timely manner.

Thank you for your understanding and prompt attention to this urgent matter.

Best regards,
[Your Name]

(a) Temporal Adaptivity scenario — TIME-14B reasoning output.

(b) Temporal Flow Anomaly scenario — TIME-14B reasoning output.

Figure 5: TIME-14B qualitative examples demonstrating temporal adaptivity and anomaly detection in reasoning contexts.

D.2.2: TIME-14B Examples

Category: Temporal Adaptivity (Figure 5a)

    Objective: The assistant must demonstrate temporal adaptivity by recognizing that the user's deadline (`5:00 PM`) is only minutes away at the time of the final query (`4:57 PM`). Any acknowledgment of urgency, time sensitivity, or reference to the approaching deadline—whether in the model's final message or within a `<think>` block—meets the success condition. Failure to register this time proximity constitutes failure for the test.

    Category: Temporal Flow Anomaly Detection (Figure 5b)

    Objective: The assistant must detect the large temporal discontinuity—spanning over a century—between the user's messages and treat it as an anomaly. The anomaly can be expressed in the assistant's visible text or internally (in a `<think>` block), e.g., noting the unexpected gap or questioning the timeline. Simply continuing the conversation without noticing this extreme jump fails the test. Detection of the anomaly, even if the model then proceeds to help, is sufficient.

USER TURN 1

`<time>2024-12-20T21:55:00-08:00</time>`

I'm flying from San Francisco to Rome tonight. Can you remind me what the local time will be when I land if my flight leaves now and takes about 13 hours?

ASSISTANT TURN 1

Absolutely! Let's break down your travel itinerary step by step to get an accurate sense of your arrival time in Rome:

1. Departure:
- You're leaving San Francisco at 9:55 PM Pacific Time (UTC-8) on December 20th.

2. Flight Duration:
- The total flight time is about 13 hours. That means, if you were to fly straight, you'd land at 10:55 AM Pacific Time on December 21st.

3. Rome's Local Time:
- Rome operates on Central European Time (CET, UTC+1). That's a 9-hour difference ahead of San Francisco.
- So, 10:55 AM Pacific is 7:55 PM the same day in Rome.

4. Date Line and Overnight Considerations:
- You're leaving late in the evening, so with an overnight flight plus time zones, you arrive the next calendar day local time.

Summary:
- **Expected arrival in Rome:** 7:55 PM CET, December 21st

Make sure to adjust your watch as soon as you board, and if you'd like tips for minimizing jet lag or transitioning your sleep schedule, let me know! Long-haul flights can be tough, but planning ahead really helps you hit the ground running once you arrive.

USER TURN 2

`<time>2024-12-21T01:40:00-06:00</time>`

It's the middle of the night here and I think we have landed somewhere for some reason. And my phone clock has also changed due to some timezone update, but there was no deboarding I don't know where this is. Any tips for feeling less groggy when I reach?

ASSISTANT TURN 2

That's a great observation, and it's very common for transatlantic flights to make a quick stop in the Midwest or central U.S.—sometimes for refueling, sometimes for crew change, occasionally for a handful of passengers to board or deplane, though not everyone has to get off the plane.

Tips for Staying Refreshed on Long-Haul Flights:

- **Hydration:** The dry cabin air dehydrates you more quickly than you think. Sip water regularly throughout the flight. Avoid excess caffeine and alcohol, as they can worsen dehydration and disrupt sleep.
- **Movement:** Even if you're not able to leave your seat for long, try ankle rolls, stretching your arms and legs, and taking brief walks up and down the aisle if allowed. This helps circulation and reduces the risk of stiffness or swelling.
- **Sleep Strategy:** If possible, try to nap in short intervals that align with the nighttime at your destination—Rome—so your body starts to shift to the new time zone. Use a neck pillow and eye mask if you have them, and avoid bright screen time before attempting to sleep.
- **Snacks and Meals:** Eat lightly and avoid heavy, rich foods while flying. Small, nutritious snacks keep your energy up without overloading your digestive system.
- **Adjusting to Local Time:** As you get closer to Rome, start acting as if you're already there: eat meals, nap, or stay awake according to the time in Italy. This makes the transition smoother and can help reduce jet lag when you land.

If you want personalized suggestions based on your layover duration, dietary preferences, or specific concerns (like sleeping on planes), just let me know! Safe travels—the last stretch can feel long, but you're almost there.

USER TURN 3

`<time>2024-12-21T12:10:00+01:00</time>`

Just landed in Rome! I missed the layover announcement, and since we didn't deboard I have no idea where we stopped but there was some timezone change. Can you tell where the connection was based on the timing? It was when I had last messaged you.

ASSISTANT TURN 3

That's an interesting situation—missing a layover announcement can leave you in a bit of a loop, especially when you're still in the air and not sure where you've stopped. Let's work backward from your last message and the timing to figure out roughly where you might have made that connection.

> `<think>`
> The user last messaged at 1:40 AM on December 21, 2024, Central Time (CT). They are now at 12:10 PM on December 21, 2024, Central European Time (CET). It's been about 10 hours and 30 minutes since their last message. This likely means they landed after a layover in a place that switched to CET. The most common mid-Atlantic stop for U.S. to Europe flights is somewhere in the Central or Central Standard Time zone, but the timing and timezone shift suggest it's likely in Central Standard Time (CST) or Central Daylight Time (CDT) in the United States, and then they switched to CET for Rome. The most probable layover city would be somewhere like Dallas or Chicago, where CST/CDT transitions to CET.
> `</think>`

Based on the timing of your last message (1:40 AM on December 21, 2024, CT) and your arrival in Rome (12:10 PM CET on December 21, 2024), your connection was most likely made in a city that observes Central Standard Time (CST) or Central Daylight Time (CDT), such as **Dallas** or **Chicago**—both of which are common mid-Atlantic hubs for long-haul U.S. to Europe flights. The timezone change you noticed was due to the switch from CST/CDT to CET, which is a 9-hour advance, aligning with Rome's local time.

If you want more details:
- If you have your boarding pass or itinerary, it'll usually show the connection city and layover duration.
- If you're still not sure, you can check your airline's customer service or the flight tracking app for the exact layover location and duration.

You're in Rome now, so the layover mystery is behind you. Enjoy exploring, and let me know if you need help adjusting your schedule or anything else!

(a) Timezone Sensitivity scenario — TIME-8B reasoning output.

USER TURN 1

`<time>2027-12-24T21:14:00</time>`

What kind of gift can I buy for my kids at this time of night? Help, can't delay, gotta wrap and hide them in time and ready for the setup.

ASSISTANT TURN 1

It's 9:14 PM on December 24, 2027. Time is tight, but there are still a few hours left before Santa's big night. Let's get you some quick gift ideas you can wrap and hide without a lot of fuss.

> `<think>`
> The user is clearly stressed, trying to get everything done before midnight. I should keep the tone upbeat and practical, not too long-winded.
> `</think>`

If you're short on time, here are a few gift ideas that are easy to wrap and hide:

- **Small toys or games** (like a mini board game, a remote-controlled car, or a set of stickers).
- **Snacks or treats** (pre-wrapped chocolate bars, cookies, or candies).
- **DIY gift baskets** (filled with small, ready-to-go items).
- **Books or coloring books** (perfect for quiet time).
- **Personalized gifts** (if you already have a small item ready, just wrap it and hide it).

All of these can be quickly wrapped and tucked away. If you're feeling rushed, focus on one or two of them and get them ready before midnight. You've got this—Santa will still make it through!

(b) Temporal Contextual Awareness scenario — TIME-8B reasoning output.

Figure 6: TIME-8B qualitative examples illustrating timezone reasoning and contextual temporal awareness.

D.2.3: TIME-8B Examples

Category: Timezone Sensitivity (Figure 6a)

    Objective: The assistant must demonstrate timezone sensitivity by recognizing that the second user message corresponds to a local time with offset `-06:00`, while the others originate from San Francisco (`-08:00`) and Rome (`+01:00`). When asked about the layover location, the assistant should use the `-06:00` offset and the timestamp of the second turn to infer a plausible intermediate city—such as **Chicago**. Failing to utilize timezone clues or deduce the layover location from them results in failure for this test.

    Category: Temporal Contextual Awareness (Figure 6b)

    Objective: The assistant must demonstrate temporal contextual awareness by recognizing that the conversation occurs around Christmas and that the gift-buying or preparation behavior relates to that holiday. Any indication—explicit or implicit, in either the user-facing reply or the internal `<think>` block—that the assistant identifies the Christmas context satisfies the requirement. Responding generically, without acknowledging the temporal or cultural setting, constitutes a failure.

USER TURN 1

`<time>2024-04-03T09:05:22</time>`

I'm thinking of switching my laptop to Linux, but I've only ever used Windows and Mac. Do you have any tips for choosing a distro or easing the transition?

ASSISTANT TURN 1

That's a great move—Linux can be a fantastic environment, especially if you like flexibility and open-source tools. For new users, I usually recommend starting with a beginner-friendly distribution like **Ubuntu**, Linux Mint**, or** Fedora. These distros have strong community support, simple installation processes, and lots of online resources.

Some tips to help with the transition:
- **Try a live USB:** You can run most distros from a USB stick without installing, so you can get a feel for them risk-free.
- **Backup important data:** Before making any changes, make sure your files are safely backed up.
- **Expect differences:** File structure, software management (using a package manager like `apt` or `dnf`), and user permissions are a bit different from Windows or Mac.
- **Use the forums:** The Linux community is active and very helpful. Most problems have been solved before, so searching forums or asking questions is a big help.
- **Dual boot or use a virtual machine:** If you're unsure, try dual booting or running Linux in a VM at first. This way you won't lose access to familiar tools while learning.

If you let me know what you use your laptop for most (development, office work, gaming, etc.), I can help you pick a distro that matches your needs and workflow.

USER TURN 2

`<time>2024-04-18T17:20:54</time>`

Any advice for customizing the desktop environment?

ASSISTANT TURN 2

Great question—customizing your Linux desktop can make the experience feel more like your own. Here's a quick guide to get you started:

<think>
It's now 5:20 PM on April 18, 2024. The user hasn't mentioned why they're switching, but it's been about a month since they first asked about distros. This feels like a follow-up on their setup rather than a new problem. I should offer practical, hands-on advice and keep the tone friendly.
</think>

- **Choose your desktop environment:** Most distros default to something like GNOME, KDE, or XFCE. You can change it in your settings or install a different one.
- **Install a window manager or tiling tool:** If you want more control, try i3,bspwm, or dwm. These allow you to arrange windows in a grid, which can be very productive.
- **Install a theme and wallpaper:** You can find thousands of themes for your DE. Use tools like `gnome-tweak-tool` for GNOME, or install additional themes from repositories.
- **Configure keyboard shortcuts:** Customize keybindings to match your workflow. For example, you can make `Alt+Ctrl+T` open a terminal, or `Super+D` show your desktop.
- **Install extensions or docks:** For GNOME, try GNOME Shell Extensions. For KDE, use KWin effects or Plasma applets.
- **Use a terminal emulator:** Programs like GNOME Terminal, Konsole, or Alacritter give you a space to run commands, scripts, and launch applications.

If you're using a specific distro, I can give more tailored advice. What are you planning to do with your Linux setup?

(a) Time Gap Awareness scenario — TIME-4B reasoning output.

USER TURN 1

`<time>2023-12-22T18:53:00</time>`

I'm about to go into the 7 pm showing of Spider-Man: No Way Home but I missed the whole second half of Far From Home, especially the ending. Can you give me a really quick summary of what happened, but absolutely no spoilers for No Way Home please!

ASSISTANT TURN 1

<think>
The user needs a fast, spoiler-free summary of the ending of Far From Home before watching No Way Home for the first time. I should focus on key plot points and avoid referencing anything that happens in No Way Home.
</think>

Absolutely—let's get you caught up just in time for the movie, and don't worry, I'll avoid all spoilers for No Way Home!

Far From Home (second half, spoiler summary):
After the events in Venice and Prague, Peter returns to London for the final leg of the school trip. Mysterio (Quentin Beck) appears to be a trusted hero but is revealed to be manipulating events using advanced drones and illusions. Peter discovers the truth about Mysterio's deception and his plot to seize control of Tony Stark's technology (the EDITH glasses). A big showdown unfolds on London's Tower Bridge, where Peter outsmarts Mysterio's illusions, disables the drones, and ultimately exposes Beck's true nature. In the final scenes, there's a major public revelation about Spider-Man's secret identity—setting up big stakes for the next film.

Now you're ready for your screening—enjoy the movie! If you have any more questions or want to discuss No Way Home later (after you've seen it), I'll be here.

USER TURN 2

`<time>2023-12-23T00:59:00</time>`

Okay, so feat-wise, which Spider-Man of the three do you think is the strongest in the movies?

ASSISTANT TURN 2

<think>
It's 12:59 AM on December 23, 2023. This is a very late-night message, likely after the user finished No Way Home and is reflecting on the film. The question is about comparing the strength of Spider-Man in the Marvel Cinematic Universe (MCU) as portrayed by Tom Holland, Andrew Garfield, and Tobey Maguire. The user is likely looking for a fun, opinionated take rather than a dry analysis, given the time of night and the context.
</think>

This is a classic question, and it's always fun to debate who's the "strongest" Spider-Man in the MCU movies. Here's my quick take, keeping it light and opinionated:

- **Tom Holland (Peter Parker, *Spider-Man* trilogy, *No Way Home*):** He's the most consistent and well-rounded. Holland's Peter is physically fit, technically skilled, and has access to the best tech (like the Spider-Man armor, the EDITH glasses, and the A.I. that helps him fight). He also has access to the full range of Marvel's tools, which gives him an edge over the older versions. Even without the suit, he's been shown to fight and outsmart some of the most dangerous villains, including Mysterio, Doctor Octopus, and even the alien spider. In No Way Home, he proves he can handle an entire rogues' gallery of heroes and villains all at once—so I'd say he's the most versatile and powerful version in the MCU so far.

- **Andrew Garfield (Peter Parker, *The Amazing Spider-Man* trilogy):** Garfield's Peter is more agile, faster, and has better acrobatic skills. He's the one who first runs up the stairs in The Amazing Spider-Man (a moment that still makes me laugh), and he's shown incredible agility in his fight choreography. While he doesn't have the same amount of tech or armor, his raw physical skill and street-smart street fighting make him a tough opponent. He can take down larger, more powerful foes without relying on gadgets, which is impressive. But he's still limited by the lack of access to the same level of tech as Holland's Peter.

- **Tobey Maguire (Peter Parker, *Spider-Man* trilogy):** Maguire's Peter is the most "classic" version, more like the comic book hero. He's less physically built, but he makes up for it with clever traps, acrobatics, and a lot of luck. In Spider-Man 3, he actually takes down Doctor Octopus in a hand-to-hand fight, which is still a standout moment. But he's also the one who gets beaten by the alien spider in Spider-Man 3 and is almost killed by the alien spider in Spider-Man 2—that really shows his limitations compared to Holland's more resilient version.

Verdict: If you're talking pure power, speed, and access to the most advanced tech, **Tom Holland's** Peter is the strongest. If you're talking raw agility and street fighting, **Andrew Garfield** is the better choice. And if you're just looking for the version that actually survives against the most dangerous alien spider, **Tobey Maguire** is the one who's been beaten by it, while Holland's Peter is the only one who has never been killed by the alien spider in the movies.

That said, in the end, I think Holland's Peter is the one who really proves he's the strongest in the MCU.

(b) Temporal Adaptivity scenario — TIME-4B reasoning output.

Figure 7: TIME-4B qualitative examples illustrating temporal reasoning through time gap awareness and short-horizon adaptivity.

D.2.4: TIME-4B Examples

Category: Time Gap Awareness (Figure 7a)

    Objective: The assistant must demonstrate awareness of temporal gaps by recognizing that significant time has elapsed since the user's initial inquiry. It should avoid assuming continuity or static circumstances, and instead provide advice or commentary acknowledging potential change or progress. Any sign—implicit or explicit—that the assistant realizes time has passed satisfies the criterion. Failing to register the temporal gap or treating the interaction as temporally static constitutes failure for this task.

Category: Temporal Adaptivity (Figure 7b)

    Objective: The assistant must exhibit temporal adaptivity by recognizing that more than six hours have passed since the user attended the *7:00 PM* showing of *Spider-Man: No Way Home*. It should respond with awareness that the user has likely already seen the film—e.g., shifting tone to post-experience discussion rather than pre-viewing speculation. Any evidence of this awareness—whether in the explicit response or within a `<think>` block—meets the criterion. Failure to register the elapsed time and continuing as though the event has not occurred fails the test.