

TIME: Temporally Intelligent Meta-reasoning Engine for Context Triggered Explicit Reasoning

Anonymous ACL submission

Abstract

Reasoning oriented large language models often expose explicit “thinking” as long, turn-global traces at the start of every response, either always on or toggled externally at inference time. While useful for arithmetic, programming, and problem solving, this design is costly, blurs claim level auditability, and cannot re-trigger explicit reasoning once the model begins presenting. Dialogue models are also largely blind to temporal structure, treating replies after seconds and replies after weeks as equivalent unless time is stated in text. We introduce TIME, the Temporally Intelligent Meta-reasoning Engine, a behavioral alignment framework that treats explicit reasoning as a context sensitive resource driven by discourse and temporal cues. TIME augments dialogue with optional ISO 8601 <time> tags, tick turns that represent silent gaps, and short <think> blocks that can appear anywhere in a reply. A four-phase curriculum including a small, maximally diverse full-batch alignment step trains Qwen3 dense models to invoke brief, in-place reasoning bursts and keep user facing text compact. We evaluate with TIMEBench, a temporally grounded dialogue benchmark probing chronology, commonsense under gaps and offsets, anomaly detection, and continuity. Across 4B to 32B scales, TIME improves TIMEBench scores over base Qwen3 in both thinking and no-thinking modes while reducing reasoning tokens by about an order of magnitude.

1 Introduction

Reasoning oriented language models increasingly make reasoning explicit by exposing thinking traces, often as chain-of-thought style text. In many systems, this reasoning appears as a long block at the start of each reply, either always on, or controlled externally through inference flags. The pattern boosts accuracy on arithmetic, programming, problem solving, and multi step planning, and it

provides a practical interface for inspecting rationale when mechanistic interpretability is unavailable. Yet it also turns explicit reasoning into a blunt instrument.

Front loaded thinking has three recurring limitations. First, it is expensive and verbose, because the model emits many reasoning tokens even for trivial or low risk requests. Second, it weakens claim level auditability. A single monologue can loosely justify many later statements, so the mapping from an individual claim to its justification is often indirect. Third, it is not reactive. Once the model begins presenting, it cannot cleanly re enter a thinking phase to re check assumptions or re anchor itself if uncertainty arises mid answer.

A better target is a reasoning policy that is closer to human thought: short bursts, inserted where needed, and re-triggered when new cues appear. Achieving this requires signals that indicate latent state change, not only a task difficulty estimate. Temporal transitions are one of the most common such signals in real interactions. Plans expire, priorities change, and users return after long gaps. Yet unless time is stated explicitly, standard dialogue models treat a reply after two seconds and a reply after two weeks as the same token sequence.

We present TIME (Temporally Intelligent Meta-reasoning Engine), a behavioral alignment framework that equips reasoning models with context sensitive explicit thinking. TIME introduces three lightweight primitives: optional <time> tags on turns, optional short <think> blocks that may appear anywhere in a response, and tick turns that contain only time metadata to represent silence and passage of time. These primitives model temporality explicitly during training and evaluation, but timestamps are not required for the learned policy to operate: the model can still trigger brief reasoning from purely textual cues when temporal metadata is absent.

Because this behavior is rare in existing corpora,

084	we align it with a four-phase curriculum on the	134
085	Qwen3 family from 4B to 32B parameters. Early	135
086	phases teach the model to parse the primitives and	136
087	to keep reasoning bursts short and well delimited.	137
088	A final phase uses a small but maximally diverse	138
089	full-batch alignment set whose only invariant is	139
090	contextually triggered reasoning keyed to temporal	140
091	and discourse cues. This pushes the model toward	141
092	a policy rather than a template, while reducing the	142
093	tendency to collapse back into verbose, turn-global	143
094	traces.	
095	To measure reasoning competence under un-	
096	certainty and transitions as well as reasoning de-	
097	ployment, we introduce TIMEBench, a temporally	
098	grounded dialogue evaluation suite that probes	
099	chronological reconstruction, temporal common-	
100	sense under gaps and timezone shifts, detection	
101	of invalid or implausible timestamps, and conti-	
102	nuity across long silences. Beyond task success,	
103	TIMEBench logs where and how often <think>	
104	blocks are used, enabling analysis of whether mod-	
105	els reason only when the situation demands it.	
106	2 Related Work	
107	Temporal Cognition and Time-Aware Modeling	
108	Prior work on time in language models primarily	
109	treats time as <i>content</i> : timestamped facts, event	
110	ordering, and knowledge that changes across time.	
111	Timestamp-conditioned modeling such as <i>Time-</i>	
112	<i>Aware LMs</i> (Dhingra et al., 2022) conditions gen-	
113	eration on temporal indices to better handle drift	
114	and time-sensitive knowledge. Complementary	
115	analyses study temporal non-stationarity and <i>tem-</i>	
116	<i>poral generalization</i> , showing that performance	
117	degrades when models are evaluated on future	
118	slices beyond their training distribution (Lazaridou	
119	et al., 2021). More recent evaluation frameworks	
120	explicitly probe time awareness over large event	
121	collections, for example TimeShift-style methods	
122	that test recall and calibration under temporal con-	
123	straints (Herel et al., 2024).	
124	Benchmarks have also expanded from synthetic	
125	temporal arithmetic toward broader temporal com-	
126	petence. ChronoSense evaluates temporal rela-	
127	tions and temporal arithmetic, while TimeE and	
128	EvolveBench test temporal reasoning under real-	
129	world dynamics, including dialogue settings, in-	
130	valid timestamp handling, and temporal misalign-	
131	ment between inputs and queries (Islakoglu and	
132	Kalo, 2025; Zhu et al., 2025; Wei et al., 2025).	
133	In parallel, retrieval and structured knowledge ap-	
	proaches incorporate temporal graphs to repre-	134
	sent evolving facts, including temporal GraphRAG-	135
	style systems for time-sensitive retrieval and up-	136
	dates (Han et al., 2025). These lines strengthen	137
	temporal factuality, ordering, and retrieval, but still	138
	focus on time as part of the <i>world state</i> . In contrast,	139
	TIME uses elapsed time and temporal discontinu-	140
	ities as <i>interaction state</i> : a ubiquitous cue for latent	141
	context shifts that should trigger explicit reasoning,	142
	re-grounding, or brevity changes.	143
	Explicit Reasoning Traces and Reasoning	144
	Control	145
	Chain-of-thought prompting (Wei et al., 2022) im-	146
	proves multi-step task performance and provides	147
	explicit reasoning traces that are often treated as op-	148
	erational explanations. However, a growing body	149
	of work shows that such traces are not reliably	150
	faithful to the computations that drive final an-	151
	swers, limiting their use as audit artefacts and mo-	152
	tivating methods that measure or improve faithful-	153
	ness (Paul et al., 2024; Tutek et al., 2025; Barez	154
	et al., 2025; Chen et al., 2025). Beyond faithful-	155
	ness, turn-global reasoning also creates practical	156
	issues: cost, latency, and weak claim-level attri-	157
	bution when a long rationale must justify many	158
	downstream statements.	159
	Most deployed systems still treat explicit rea-	160
	soning as a mode rather than a policy. Hybrid rea-	161
	soners allow inference-time toggles, but the model	162
	itself does not decide when to reason. Recent work	163
	begins to learn this decision, for example by train-	164
	ing hybrid-reasoning models to select whether to	165
	think, often using reinforcement learning or con-	166
	troller objectives (Jiang et al., 2025). Other ap-	167
	proaches control <i>how much</i> a model reasons via	168
	length-conditioned policy optimization (Aggarwal	169
	and Welleck, 2025). Orthogonally, methods such	170
	as DOTS search over reasoning action trajectories,	171
	and RL-based interleaving trains models to alter-	172
	nate thinking and answering to reduce time-to-first-	173
	token and improve efficiency (Yue et al., 2025; Xie	174
	et al., 2025). These advances show progress to-	175
	ward adaptive reasoning, but are typically framed	176
	around task difficulty, length budgets, or action	177
	planning, and often remain effectively turn-global	178
	at inference time. TIME targets a different axis:	179
	short, in-place reasoning bursts triggered by sit-	180
	uational cues, including temporal transitions and	181
	discourse discontinuities, enabling mid-response	182
	re-anchoring when uncertainty or drift emerges.	183

3 Framework

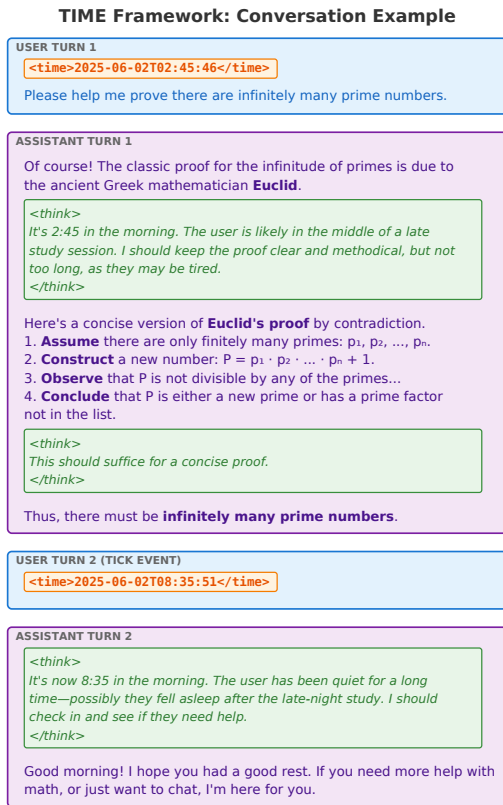


Figure 1: A TIME conversation sample.

Overview

TIME aligns reasoning models to treat explicit thinking traces as a *context-sensitive resource*. Instead of emitting one turn-global reasoning block at the start of every reply, the model learns when to expose short reasoning bursts and can re-anchor mid-response. Temporal transitions are a strong and common source of latent context change, but the learned policy is not limited to time and can also trigger from non-temporal cues in text.

Primitives

TIME introduces three lightweight primitives that can be embedded in chat logs (Table 1): optional timestamps, optional transient `<think>` blocks, and tick turns that represent silent time advance.

Timestamped Turns (`<time>`)

A user turn may include a timestamp tag:

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

The `<time>` tag encodes interaction time so the model can infer elapsed intervals and re-

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

Table 1: TIME primitives that structure behavior during training and evaluation.

engagement after gaps, which are otherwise invisible in token-only context. Timestamps are not mandatory at inference; they mainly make temporality explicit during training and evaluation. After alignment, the same policy can still trigger `<think>` blocks from textual cues such as uncertainty, contradictions, task difficulty or goal changes.

Transient Reasoning (`<think>`)

TIME uses `<think>` blocks as explicit thinking traces that are **short, optional, and placeable**. A response may include zero or multiple `<think>` blocks, and blocks may appear anywhere in the turn. This supports selective invocation and re-triggering during presentation, for example to verify a claim, recover context, or respond to a cue that appears mid-answer. For auditability, blocks are trained to be bounded and decision-linked rather than full solution transcripts.

Tick Events

A **tick** is a user turn containing only a timestamp:

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

Ticks simulate time passing without new text and provide a controlled way to model persistence and delayed follow-up. During training, ticks teach the model to treat non-interaction as context and to decide whether to stay silent, check in briefly, or refresh a plan after a long gap. Ticks are not required at inference; they serve as a modeling primitive and a stress-test for reasoning policies under temporal discontinuity.

Temporal Behavior as Sequence Control

Together, `<time>`, `<think>`, and ticks turn dialogue into a temporally anchored behavioral sequence. The model learns when to think, when to speak, and how much reasoning to expose using temporal and non-temporal cues. Figure 1 shows a representative thread with a silent gap and a follow-up that acknowledges elapsed time with adjusted

244	tone and brevity. Further examples are included in		
245	Appendix B .		
246	4 Methods		
247	4.1 Curriculum and Alignment Protocol		
248	We align TIME via a staged, curriculum-based		
249	supervised fine-tuning (SFT) procedure on the		
250	Qwen3 family at four sizes (32B, 14B, 8B, 4B).		
251	Training uses QLoRA (Detrmers et al., 2023) on a		
252	single NVIDIA RTX Pro 6000 Blackwell (96GB) .		
253	The objective is a reasoning invocation policy :		
254	emit <think> traces only when cues warrant them,		
255	keep traces brief and well-delimited, and allow re-		
256	triggering later in the same response if warranted.		
257	Why Qwen3 and why curriculum. Qwen3 is a		
258	strong hybrid reasoner, but it is mode-bound: think-		
259	ing mode tends toward long, front-loaded <think>		
260	blocks, while no-thinking mode emits none. Exter-		
261	nal toggles do not teach the model when to reason		
262	or where to place reasoning. TIME targets a policy		
263	that is optional, brief, and reactive to cues such		
264	as uncertainty, instruction conflicts, and time gaps.		
265	Direct SFT onto the final behavior is unstable and		
266	often collapses back to templated or verbose traces,		
267	so we introduce prerequisites first and then consol-		
268	idate with a small, high-entropy alignment set.		
269	Shared configuration for Phases 1 to 3. Phases		
270	1 to 3 share a fixed recipe: LoRA rank 32 on projec-		
271	tion layers ($\alpha = 32$, dropout 0.05); AdamW-8bit		
272	with learning rate 2×10^{-5} and linear decay; effec-		
273	tive batch size 32; gradient checkpointing; and 3		
274	epochs per phase. Each phase replays 25% of prior		
275	data. About 5% of samples include system prompts		
276	describing the protocol with varied wording.		
277	Synthetic data for the first three phases was gener-		
278	ated using a template-guided process seeded by		
279	hand-curated examples on GPT-4o and Gemini		
280	2.5 Flash, followed by automated sanity checks.		
281	The stylistic variety is intentionally narrow; these		
282	phases teach parsing of primitives, local placement		
283	of reasoning, and length control.		
284	Phase 1: Structural Seeding		
285	(2,188 train / 387 test) Single-turn prompts teach		
286	the markup: <time> metadata plus short <think>		
287	bursts for compact intermediate steps. The goal		
288	is to normalize brief, well-delimited traces and		
289	weaken the preference for long, monolithic rea-		
290	soning.		
	Phase 2: Temporal Exposure		291
	(4,745 train / 838 test \rightarrow 5,291 / 935 with re-		292
	play) Two-turn dialogues introduce time gaps and		293
	the first tick events. The model must condition on		294
	temporal metadata to revise assumptions and re-		295
	establish context after silence. This phase also in-		296
	troduces early pressure against reflexive verbosity,		297
	requiring the model to infer when silence is suffi-		298
	cient or when a concise contextual update is more		299
	appropriate.		300
	Phase 3: Contextual Modulation		301
	(4,147 train / 732 test \rightarrow 5,878 / 1,039 with re-		302
	play) Multi-turn dialogues teach suppression and		303
	re-triggering of <think> blocks while adjusting		304
	tone and verbosity. Tick frequency increases (about		305
	33%), encouraging the model to rely on temporal		306
	and non-temporal cues. This phase creates behav-		307
	ioral headroom for the final alignment.		308
	Phase 4: Gradient-Aligned Convergence via		309
	Maximal Diversity		310
	(128 hand-curated conversations, no replay)		311
	Phase 4 is the decisive alignment step. We use a		312
	small but maximally diverse set whose only shared		313
	property is the target policy: <think> bursts are-		314
	triggered by temporal or discourse cues and placed		315
	where needed. Each conversation is multi-turn, so		316
	supervision covers many state transitions, including		317
	long-gap re-engagement and tick-driven silence.		318
	We disable replay to avoid reintroducing earlier		319
	regularities.		320
	Why 128 examples. The goal is to align a pol-		321
	icy, not to expand knowledge. Full-batch updates		322
	require the effective batch size to equal the dataset		323
	size, and this behavior has a narrow stability win-		324
	dow: over-optimization leads to looping, <think>		325
	bleed, and style collapse. A compact, high-entropy		326
	batch concentrates gradients on the intended invari-		327
	ant while remaining stable.		328
	Deterministic Full-Batch Fine-Tuning		329
	We train with effective batch size 128, so each		330
	update sees the entire alignment set, removing sam-		331
	pling variance. Table 3 reports configuration.		332
	Across all four scales, we observe an abrupt		333
	degeneracy tipping-point : once training loss dips		334
	below ≈ 1.04 , the frequency of infinite loops,		335
	<think> format bleed, and style collapse rises		336
	sharply. We therefore retain the last clean epoch —		337
	18 / 25 / 30 / 31 for the 32B / 14B / 8B /		338

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, tick events after long gaps.
Noise	Typos, abrupt topic shifts, contradictory instructions, diverging user preferences.

Table 2: Maximal surface diversity in Phase 4 isolates the intended behavioral invariant.

Setting	Value
Effective Batch Size	128 (entire dataset)
Optimizer	AdamW-8bit
LR / warm-up	1.5×10^{-4} , 6 warm-up steps
Epochs scanned	35 / 36 / 40 / 46 (32B / 14B / 8B / 4B)
Checkpoint criterion	First epoch with loss < 1.05 and no sharp rise in degeneracy
LoRA	Rank 32, $\alpha = 32$

Table 3: Phase 4 tuning setup. Clean convergence is assessed using degeneracy suppression.

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

Phase 4 is structured so that the only gradient-aligned invariant is the reasoning policy itself, encoded through a maximally diverse batch of 128 samples. The model must detect contextual cues such as elapsed time, tick events, contradictions, or discourse shifts, decide whether to emit a bounded <think> burst and where to place it, and keep the user-facing presentation concise. Unlike mini-batch training, where sampling variance can allow incidental correlations (e.g., topic or formatting artifacts) to dominate updates, full-batch alignment over the entire diverse set induces a regularizing effect that suppresses such spurious signals, yielding the final checkpoints used for evaluation, henceforth referred to as **TIME-32B / 14B / 8B / 4B**.

Further training details are documented in [Appendix C](#).

4.2 Evaluation Method: TIMEBench

TIMEBench is a 77-scenario benchmark for **temporal common sense reasoning in dialogue**. We use temporality because it is one of the most frequent drivers of latent context transitions in real interaction: deadlines move, users disappear and return, assumptions go stale, and coordination depends on time and timezone. These transitions are often invisible in token-only chat logs unless time is represented explicitly. TIMEBench therefore provides a general-purpose test of whether a model can update beliefs and behavior under temporal uncertainty, regardless of whether the model is an instruction model, a thinking model, or a hybrid reasoner. TIMEBench makes these transitions explicit and tests whether a model responds consistently.

Structure. The benchmark covers **seven diagnostic categories** with **11 scenarios each**:

- **Chronological Retrospection**¹: reconstruct timeline facts across turns.
- **Invalid Time Detection**¹: detect impossible dates or times.
- **Temporal Adaptivity**²: adjust urgency and verbosity near deadlines.
- **Temporal Contextual Awareness**²: infer plausible user context from time cues.
- **Temporal Flow Anomaly Detection**¹: detect implausible jumps or reversals in time.
- **Time Gap Awareness**²: revise assumptions after long but plausible silence.
- **Timezone Sensitivity**¹: interpret meaning implied by changes in UTC offsets.

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

²Category reflects curriculum-intended behavior, though all scenarios remain unseen during training.

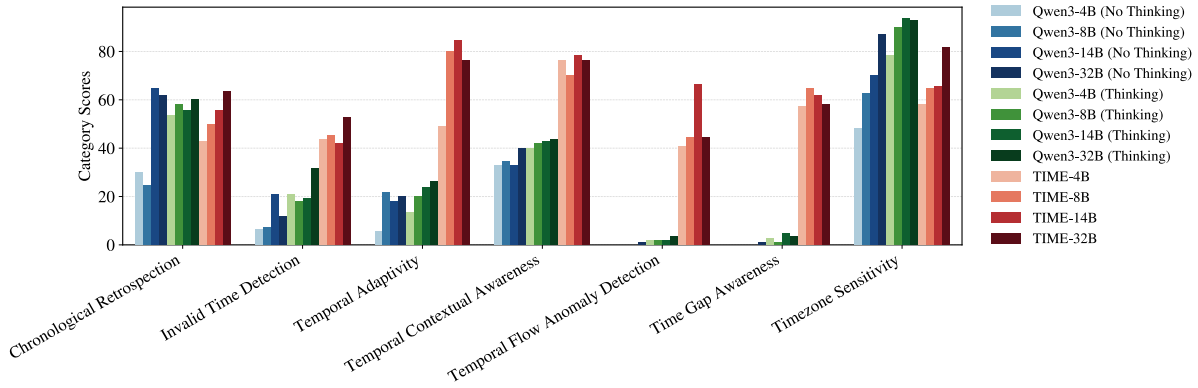


Figure 2: Breakdown of TIMEBench scores by diagnostic category and model size.

Representative examples for each diagnostic category are provided in the appendix (Appendix E).

Scenario design. Each scenario is a short, naturalistic dialogue thread with timestamps per user query; the model generates only the final assistant turn. Every scenario is paired with a category-specific objective that defines the behavior being tested (e.g., detecting implausible timestamps or adjusting assumptions after a long gap). Objectives focus on everyday temporal plausibility rather than rare trivia, making scores meaningful for any LLM, including non-reasoners, fixed reasoners, and hybrid models in either thinking or no-thinking mode.

Protocol. Each model is evaluated with **10 trials per scenario** using PCG64-derived seeds (770 runs total). Trials are scored with **binary objectives** (0 or 1) by a third-party LLM judge that sees only the model response and the objective, not the original prompt or timestamps. We aggregate trial scores to scenario scores (mean of 10), then category scores (mean over scenarios, scaled to %), and an overall TIMEBench score (mean over categories). We estimate **95% confidence intervals** via stratified bootstrapping (10,000 resamples) by resampling scenario scores within each category and recomputing all aggregates.

Behavioral instrumentation. In addition to correctness, TIMEBench enables a structural audit of generation. We extract: whether `<think>` appears, its position (start, mid, end), mean reasoning and output token counts, light versus heavy markdown usage, format bleed, and degeneracy indicators including infinite repetition. This second view lets us study how context-triggered thinking policies change temporal commonsense performance and which structural shifts co-occur with gains.

5 Results

Size	Qwen3		TIME
	No-Thinking	Thinking	
4B	17.53	30.13	52.60
8B	21.56	32.99	59.87
14B	29.48	34.42	64.80
32B	31.82	37.40	64.81

Table 4: TIMEBench scores (out of 100) across models.

We evaluate all models on **TIMEBench**, comparing our aligned models (**TIME-4B / 8B / 14B / 32B**) to the corresponding base Qwen3 checkpoints in both no-thinking mode (via the `/no_think` suffix) and full thinking mode. All models use identical decoding parameters: **temperature 0.6, top-p 0.95, top-k 20, min-p 0**, following the reasoning evaluation recommendations in the *Qwen3 Technical Report* (Qwen-Team, 2025). GPT-5.2 (2025-12-11 checkpoint) accessed via the OpenAI API serves as the blind judge model for all binary evaluations.

Table 4 reports aggregate **TIMEBench scores (0 to 100)**. As visualized in Figure 2, TIME models consistently outperform both Qwen3 baselines across all four sizes. The gains are large and stable: for example, at 4B, TIME exceeds Qwen3 (Thinking) by more than 22 points (52.60 vs. 30.13), and even at 32B the aligned model improves over the thinking baseline by more than 27 points (64.81 vs. 37.40). Similar gaps appear at 8B and 14B, where TIME approaches or surpasses the temporal commonsense performance of much larger base models, despite using the same underlying weights.

Bootstrapped **95% confidence intervals** confirm that these trends are not driven by a small subset of scenarios. For TIME-4B the interval spans

Model	Benchmark Score	Runs w/ <think> (%)	Mean Number of <think>	Think Position			Mean Number of Tokens		Runs w/ Degeneracy (%)
				Start (%)	Mid (%)	End (%)	Thinking	Output	
No-Thinking	31.82	00.0	0.00	—	—	—	0.00	608.96	4.42
Thinking	37.40	99.2	0.99	100.0	0.0	0.0	910.52	1573.47	18.18
Phase 1	42.47	99.5	0.99	100.0	0.0	0.0	803.52	1434.56	13.90
Phase 2	56.88	95.6	1.12	70.7	29.1	0.2	76.59	362.45	4.68
Phase 3	52.08	89.2	1.25	55.0	44.6	0.4	52.94	294.51	0.26
TIME	64.81	80.6	1.67	24.1	75.6	0.2	84.16	332.64	3.64

Table 5: Phase-wise ablation: Structural and behavioral metrics for Qwen3-32B across curriculum stages and final alignment to TIME-32B

44.55 to 60.39, for TIME-8B 53.38 to 66.23, for TIME-14B 59.09 to 70.39, and for TIME-32B 58.18 to 71.17. By comparison, Qwen3 in thinking mode remains substantially lower at every size: Qwen3-4B ranges from 23.90 to 36.36, Qwen3-8B from 26.88 to 39.09, Qwen3-14B from 28.44 to 40.65, and Qwen3-32B from 31.56 to 43.51. No confidence interval for a TIME model overlaps the corresponding thinking baseline.

To test robustness at the scenario level, we apply a **Wilcoxon signed-rank test** to per-scenario average scores, comparing TIME against Qwen3 (Thinking) for each size. Improvements are statistically significant in all cases ($p < 0.001$): for 4B, $p = 3.8e-4$; for 8B, $p = 1.9e-5$; for 14B, $p = 1.6e-6$; and for 32B, $p = 5.0e-7$. These results indicate that TIME delivers consistent gains in temporal commonsense reasoning across categories and scales, in line with the hypothesis that a context-triggered explicit reasoning policy helps models handle temporal state transitions more reliably than static thinking modes. A full category-level and structural breakdown across model sizes is provided in [Appendix F](#).

5.1 Phase-wise Ablation in 32B: Diagnostic and Structural Metrics

[Table 5](#) separates two dimensions that are often conflated in reasoning model evaluations. The **score** column compresses TIMEBench category accuracies into a single proxy for commonsense robustness under state changes such as temporal gaps, anomalous inputs, and offset shifts. The remaining columns capture **structural behavior**: whether explicit reasoning is used, where it appears within a turn, how many tokens it consumes, and how often decoding degenerates. Together with [Figure 3](#), this allows us to study how a context-triggered reason-

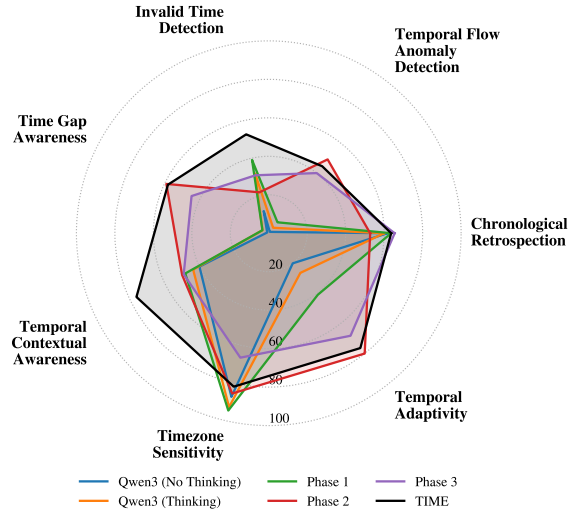


Figure 3: Evolution of core temporal competencies in Qwen3-32B across curriculum phases.

ing policy reshapes generation.

Baselines. The no-thinking baseline handles local cues well, with strong timezone sensitivity and reasonable short-range retrospection, but fails on transition-oriented categories such as flow anomaly detection and gap awareness and misses most invalid timestamps. Enabling full thinking improves scrutiny with significant rise in invalid time detection, yet transition categories remain near floor. Structurally, the thinking baseline behaves like a classic chain-of-thought model: runs contain a single front-loaded trace, averaging about 911 reasoning tokens and 1,573 total output tokens, with a high degeneracy rate of 18.18%.

Phase 1 (structural seeding). Phase 1 primarily teaches the model to operate within the new syntax. Scores increase modestly, but the reason-

ing policy remains unchanged. Almost all runs still use a single preamble-style trace, with long explicit reasoning (about 804 tokens) and large outputs (about 1,435 tokens). Degeneracy drops to 13.90%, largely due to reduced formatting errors.

Phase 2 (temporal exposure). Phase 2 marks the first point where diagnostic and structural metrics shift together. Categories probing reasoning under transitions and anomalous flows, including adaptivity, flow anomaly detection, and gap awareness, rise sharply, indicating that temporal discontinuities are now treated as informative cues. At the same time, the explicit reasoning budget collapses to about 77 tokens per run and output length to about 362 tokens. Mid-turn `<think>` blocks appear in roughly 29% of runs, and degeneracy drops to 4.68%. Explicit reasoning is no longer a fixed preamble but a resource invoked by context.

Phase 3 (contextual modulation). Phase 3 emphasizes restraint and stability. Chronological retrospection and adaptivity remain strong while the reasoning budget falls further to about 53 tokens and output length to about 295 tokens. Degeneracy reaches its minimum at 0.26%, and mid-turn reasoning becomes more common than front-loaded traces. However, some transition-sensitive categories lose part of their Phase 2 gains, suggesting that suppression pressure can prevent useful re-anchoring when subtle discontinuities warrant it.

TIME (Phase 4 alignment). Final alignment in TIME-32B restores robustness on transition and anomaly categories while preserving intermittent reasoning. It achieves the best overall score, uses only about 84 reasoning tokens and 333 output tokens per run, and emits `<think>` blocks in roughly 81% of runs, now predominantly mid-turn (75.6%) rather than front-loaded (24.1%). Degeneracy remains low at 3.64%. This profile matches the intended behavior: explicit reasoning is neither always-on nor absent, but concentrated where temporal or contextual shifts make commonsense reasoning most brittle. Fine-grained diagnostics are reported in [Appendix F](#).

6 Discussion

The results support viewing explicit reasoning as a learned control policy rather than a fixed decoding style. Under TIME, Qwen3 shifts from long, always on chains of thought to compact reasoning bursts that are mostly triggered by state changes,

such as looming deadlines, long silences, or inconsistent timestamps. The strongest TIMEBench gains appear in categories that require detecting such shifts and adapting the response, which suggests that the model is tracking conversational state instead of only recalling temporal facts.

Structural metrics reinforce this interpretation. Mid turn think blocks, much smaller reasoning token budgets, and lower degeneracy show that the model is more selective about when it spends explicit computation and more careful about how it separates thought from presentation. This selective allocation matters operationally: it reduces latency and token cost when fast responses are needed, while still enabling explicit reasoning to surface when the interaction becomes uncertain, discontinuous, or high stakes. The same behavior is well aligned with real time and agentic settings, where an assistant must interleave quick actions with occasional deliberate reflection rather than committing to a single global reasoning mode.

Importantly, the learned controller is not tied to explicit timestamps at inference time. When temporal metadata is present it is used as an additional cue, but the policy also operates on purely textual context and task description. Together, these observations point toward a design pattern where compact alignment sets and behaviorally targeted curricula shape how large models allocate and expose reasoning in response to changing context.

7 Conclusion

TIME introduces a lightweight framework for contextual reasoning in dialogue models. By combining a simple protocol for timestamps, ticks, and optional think blocks with a staged curriculum and a diversity driven full-batch alignment step, we convert a fixed chain-of-thought backbone into a family of models that deploy reasoning selectively, in short bursts, and in response to state changes rather than by default.

TIMEBench provides the matching evaluation lens. It shows consistent gains in temporal commonsense across seven diagnostic categories and reveals the accompanying structural shift toward contextual reasoning, smaller thinking budgets, and fewer degenerate outputs. TIME and TIMEBench together illustrate how carefully designed alignment interventions can change not only how large models answer, but how and when they choose to explicitly reason.

611 Limitations

612 While TIME provides a practical recipe for con-
613 text triggered reasoning, its scope and claims are
614 deliberately narrow.

615 All experiments use dense hybrid reasoners from
616 the Qwen3 family, which already support both in-
617 struct style and explicit reasoning modes. This
618 makes them a natural first target for learning a finer
619 grained reasoning policy, but it also limits gener-
620 ality. We do not claim that the same curriculum
621 and alignment recipe will transfer without modifi-
622 cation to purely instruct models such as Llama 3
623 Instruct variants that have not been pre trained for
624 explicit thinking traces. In such models the bottle-
625 neck may be the reasoning substrate rather than the
626 controller, and additional pre training or larger su-
627 pervised traces may be required before a policy like
628 TIME is effective. We also have not tested TIME on
629 mixture of experts hybrid reasoners such as GLM
630 4.5 Air, where routing and capacity considerations
631 may require different adapter placement or larger
632 alignment sets.

633 TIMEBench focuses on temporal commonsense
634 and conversational robustness under state transi-
635 tions. It is not a task benchmark and does not
636 measure domain specific skills such as mathemati-
637 cal problem solving, coding, or multi step tool use.
638 In such settings, traditional more verbose reason-
639 ing frameworks, including long chain-of-thought
640 or tree structured methods, may still provide higher
641 raw task accuracy than the short bursts encouraged
642 by TIME. We target situations where compact, in-
643 termittent traces are preferred over maximal per-
644 formance, for example in interactive or resource
645 constrained deployments, and therefore do not eval-
646 uate TIME on standard coding, math, or tool bench-
647 marks.

648 Our alignment protocol is entirely supervised
649 and adapter based. The final behavior is induced
650 through a small, hand curated full-batch dataset that
651 encodes a particular style of intermittent reasoning
652 and temporal sensitivity. We do not explore rein-
653 forcement learning or bandit style objectives that
654 explicitly optimise a trade off between accuracy,
655 latency, and reasoning cost, nor do we combine
656 TIME with reward models that penalise unfaithful
657 or gratuitous traces. The 128 example full-batch
658 alignment stage is also a methodological choice.
659 We argue that its small size and high diversity help
660 isolate a single behavioral invariant, but we only
661 demonstrate this on Qwen3 and under one protocol;

662 other base models or more complex settings may
663 require larger or differently structured alignment
664 sets.

665 TIMEBench is a new benchmark created along-
666 side the framework. Although we separate training
667 and evaluation data and organise scenarios by diag-
668 nostic category, the benchmark is not independent
669 in the sense of being designed by a separate group
670 and covers only a subset of time dependent rea-
671 soning phenomena. Future work should compare
672 TIME tuned models on other temporal benchmarks
673 and on general purpose evaluation suites.

674 All TIMEBench scores are obtained with an
675 LLM judge rather than human annotators. The
676 judge is blind to prompts and timestamps and
677 we use binary scoring, multiple trials, and boot-
678 strapped confidence intervals to reduce variance,
679 but this does not remove all potential biases of the
680 judging model. We do not report human studies on
681 perceived faithfulness, usefulness, or trust in the
682 produced traces.

683 Our experiments are conducted in English and
684 we do not investigate multilingual behavior. There
685 is no guarantee that the same curriculum and align-
686 ment signals would transfer to languages with dif-
687 ferent tense systems, temporal conventions, or dis-
688 course norms. We also do not evaluate safety,
689 bias, or fairness effects of the new reasoning policy.
690 While the method reduces some forms of degener-
691 acy and verbosity, it is not a safety mechanism and
692 should not be treated as such.

693 Finally, TIME addresses only one dimension of
694 model behavior, namely when and how explicit rea-
695 soning is surfaced in response to contextual cues. It
696 does not provide mechanistic interpretability, and
697 it does not constrain or verify the internal compu-
698 tation that leads to the traces. It should be viewed
699 as a step toward more auditable models, not as a
700 complete solution to transparency or accountability
701 in high stakes deployments.

702 References

- 703 Pranjali Aggarwal and Sean Welleck. 2025. L1:
704 Controlling how long a reasoning model thinks
705 with reinforcement learning. *arXiv preprint*
706 *arXiv:2503.04697*.
- 707 Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael
708 Lan, Vincent Wang, Noah Siegel, Nicolas Collignon,
709 Clement Neo, Isabelle Lee, Alasdair Paren, and
710 Yoshua Bengio. 2025. [Chain-of-thought is not ex-
711 plainability](#). *Oxford Martin AI Governance Initiative*
712 *Working Paper*.

713	Yanda Chen, Joe Benton, Ansh Radhakrishnan,	Martin Tutek, Fateme Hashemi Chaleshtori, Ana	768
714	Jonathan Uesato, Carson Denison, John Schulman,	Marasovic, and Yonatan Belinkov. 2025. Measuring	769
715	Arushi Somani, Peter Hase, Misha Wagner, Fa-	chain-of-thought faithfulness by unlearning reason-	770
716	bien Roger, and 1 others. 2025. Reasoning models	ing steps. In <i>Proceedings of the 2025 Conference on</i>	771
717	don't always say what they think. <i>arXiv preprint</i>	<i>Empirical Methods in Natural Language Processing</i>	772
718	<i>arXiv:2505.05410</i> .	(<i>EMNLP</i>).	773
719	Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and	Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten	774
720	Luke Zettlemoyer. 2023. Qlora: efficient finetuning	Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le,	775
721	of quantized llms. In <i>Proceedings of the 37th Interna-</i>	and Denny Zhou. 2022. Chain-of-thought prompt-	776
722	<i>tional Conference on Neural Information Processing</i>	ing elicits reasoning in large language models. In	777
723	<i>Systems, NIPS '23, Red Hook, NY, USA. Curran</i>	<i>Advances in Neural Information Processing Systems,</i>	778
724	<i>Associates Inc.</i>	volume 35, pages 24824–24837.	779
725	Bhuwan Dhingra, Jeremy R. Cole, Julian Martin	Shaohang Wei, Wei Li, Feifan Song, Wen Luo, Tianyi	780
726	Eisenschlos, Daniel Gillick, Jacob Eisenstein, and	Zhuang, Haochen Tan, Zhijiang Guo, and Houfeng	781
727	William W. Cohen. 2022. Time-aware language mod-	Wang. 2025. TimE: A multi-level benchmark for	782
728	els as temporal knowledge bases . <i>Transactions of the</i>	temporal reasoning of LLMs in real-world scenar-	783
729	<i>Association for Computational Linguistics</i> , 10:257–	ios. <i>arXiv preprint arXiv:2505.12891</i> . (Authors	784
730	273.	corrected from Guo et al.; First author is S. Wei).	785
731	Jiale Han, Austin Cheung, Yubai Wei, Zheng Yu,	Roy Xie, David Qiu, Deepak Gopinath, Dong Lin, Yan-	786
732	Xusheng Wang, Bing Zhu, and Yi Yang. 2025. RAG	chao Sun, Chong Wang, Saloni Potdar, and Bhuwan	787
733	meets temporal graphs: Time-sensitive modeling and	Dhingra. 2025. Interleaved reasoning for large lan-	788
734	retrieval for evolving knowledge. <i>arXiv preprint</i>	guage models via reinforcement learning. <i>arXiv</i>	789
735	<i>arXiv:2510.13590</i> .	<i>preprint arXiv:2505.19640</i> .	790
736	Jaromír Herel, Milan Straka, and Ondřej Bojar. 2024.	Murong Yue, Wenlin Yao, Haitao Mi, Dian Yu, Ziyu	791
737	Time awareness in large language models . <i>arXiv</i>	Yao, and Dong Yu. 2025. DOTS: Learning to reason	792
738	<i>preprint arXiv:2409.13338</i> .	dynamically in LLMs via optimal reasoning trajec-	793
739	Duygu Sezen Islakoglu and Jan-Christoph Kalo. 2025.	tories search . In <i>Proceedings of the International</i>	794
740	ChronoSense: Exploring temporal understanding in	<i>Conference on Learning Representations (ICLR)</i> .	795
741	large language models with time intervals of events.	Zhiyuan Zhu, Yusheng Liao, Zhe Chen, Yuhao Wang,	796
742	In <i>Proceedings of the 63rd Annual Meeting of the As-</i>	Yunfeng Guan, Yanfeng Wang, and Yu Wang. 2025.	797
743	<i>sociation for Computational Linguistics (ACL)</i> , vol-	EvolveBench: A comprehensive benchmark for as-	798
744	ume 2. Association for Computational Linguistics.	sessing temporal awareness of language models. In	799
745	Short Papers.	<i>Proceedings of the 63rd Annual Meeting of the Asso-</i>	800
746	Lingjie Jiang, Xun Wu, Shaohan Huang, Qingxiu Dong,	<i>ciation for Computational Linguistics (ACL)</i> .	801
747	Zewen Chi, Li Dong, Xingxing Zhang, Tengchao		
748	Lv, Lei Cui, and Furu Wei. 2025. Think only when		
749	you need with large hybrid-reasoning models. <i>arXiv</i>		
750	<i>preprint arXiv:2505.14631</i> .		
751	Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gri-		
752	bovskaya, Devang Agrawal, Adam Liska, Tayfun		
753	Terzi, Mai Gimenez, Cyprien de Masson d'Autume,		
754	Tomas Kocisky, Sebastian Ruder, Dani Yogatama,		
755	Kris Cao, Susannah Young, and Phil Blunsom. 2021.		
756	Mind the gap: Assessing temporal generalization		
757	in neural language models. In <i>Advances in Neural</i>		
758	<i>Information Processing Systems</i> , volume 34, pages		
759	29348–29363.		
760	Debjit Paul, Robert West, Antoine Bosselut, and Boi		
761	Faltings. 2024. Making reasoning matter: Measur-		
762	ing and improving faithfulness of chain-of-thought		
763	reasoning . In <i>Findings of the Association for Com-</i>		
764	<i>putational Linguistics: EMNLP 2024</i> , pages 15012–		
765	15032. Association for Computational Linguistics.		
766	Qwen-Team. 2025. Qwen3 technical report . <i>Preprint</i> ,		
767	<i>arXiv:2505.09388</i> .		

802	Appendix Overview		
803	This appendix contains supplementary material that	All training, statistical analysis, and inference were	851
804	supports the claims, methodology, and empirical	performed on a single machine with the following	852
805	results presented in the main paper. The material	hardware and system configuration:	853
806	is organized to emphasize reproducibility first, fol-		
807	lowed by qualitative and quantitative analyses.		
808	• Appendix A: Reproducibility Infrastructure	• CPU: AMD Ryzen 9 7950X3D	854
809	Structure	• Memory: 128 GB DDR5	855
810	Describes the experimental infrastructure re-	• GPU: NVIDIA RTX Pro 6000 Blackwell (96	856
811	quired to reproduce all results in the paper,	GB VRAM)	857
812	including hardware and software configura-	• Operating System: Ubuntu 24.04.3 LTS (in-	858
813	tions.	side WSL2 , Windows 11 Build 26100)	859
814	• Appendix B: Representative Full-	• CUDA Version: 13.0	860
815	Conversation Examples	• Driver Version: NVIDIA 582.08	861
816	Presents complete multi-turn conversations	• NVIDIA-SMI: 580.118	862
817	generated by TIME-32B on novel temporal	• Compute Capability: ≥ 12.0	863
818	reasoning prompts that were not seen during	• Python: 3.12.12	864
819	training or evaluation. These examples		
820	illustrate context-sensitive behavior.	All experiments were conducted on single-node,	865
821	• Appendix C: Training Details	single-GPU setups without distributed training.	866
822	Details the full training pipeline, including		
823	curriculum construction, dataset composition	Python Environments	867
824	and statistics, and fine-tuning configurations	Two separate environments were used due to pack-	868
825	across all phases.	age compatibility requirements:	869
826	• Appendix D: Evaluation Methodology	A.0.1 Training & Statistical Analysis	870
827	Specifies the evaluation procedures for the	Used for all fine-tuning phases, ablations, and post-	871
828	TIMEBench benchmark, including the scor-	evaluation analyses:	872
829	ing pipeline, prompt sampling configuration,	torch==2.9.1	873
830	and evaluator model setup. Documents im-	unsloth==2025.12.8	874
831	plementation details for structural behavior	transformers==4.57.3	875
832	analysis, reasoning token estimation, mark-	accelerate==1.12.0	876
833	down compliance assessment, and confidence	numpy==1.26.4	877
834	interval estimation via stratified bootstrap.	xformers==0.0.33.post2	878
835	• Appendix E: TIMEBench Completion Ex-	scipy==1.16.3	879
836	amples	A.0.2 Inference (vLLM Hosting)	880
837	Provides high-scoring completions from	Used for all inference-time runs, including	881
838	TIMEBench , with two representative exam-	TIMEBench:	882
839	ples per model size. For each scenario, only	torch==2.9.0	883
840	the final model turn is generated; all preced-	vllm==0.13.0	884
841	ing turns are fixed and authored as part of the	transformers==4.57.3	885
842	benchmark specification.		
843	• Appendix F: Detailed Ablations and	Each environment was version-locked and iso-	886
844	Metrics	lated to ensure reproduction and non-interference	887
845	Reports comprehensive diagnostic	between toolchains (e.g., Unsloth-based fine-	888
846	metrics—both descriptive and boot-	tuning vs. vLLM inference streaming).	889
847	strapped—across model sizes and training		
848	variants (e.g., with and without the TIME		
849	curriculum).		

B Representative Full-Conversation Examples

The following are real conversations generated by **TIME-32B** on scenarios **not included in training or evaluation** (that is, they do not appear in any curriculum phase or in TIMEBench).

These samples were chosen from actual usage to illustrate the kinds of **context triggered reasoning** and **temporal commonsense** that our framework encourages. In particular, they highlight:

- Situational anchoring when context evolves over time or across turns
- Shifts in tone, planning, and verbosity as the conversation state changes
- Structured, bounded reasoning with `<think>` blocks used as short bursts
- Adaptive formatting and presentation that responds to dialogue style and urgency

These examples are intended as **qualitative anchors** to help readers understand how the learned reasoning policy behaves in realistic settings, complementing the aggregate scores reported in the main text.

Concretely, the conversations show:

- How and when the model chooses to invoke `<think>` blocks, including mid turn
- How contextual cues, silences, or shifting constraints change the structure and intent of the reply
- How the framework can still function without `<time>` blocks in input
- How explicit reasoning traces can function as compact audit artefacts attached to specific recommendations

Taken together, these conversations provide an intuitive feel for the **emergent structure** that TIME scaffolds through its curriculum and alignment protocol, and for the kind of contextual reasoning behavior that TIMEBench is designed to diagnose.

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 with context triggered reasoning)

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 with temporal state shifts)

USER TURN 1
 Three boxes sit on a table: one red, one blue, and one green. Each box contains exactly one item: a coin, a key, or a ring. The red box does not contain the coin. The blue box contains neither the ring nor the key. The green box does not contain the ring. Question: Which item is in each box?

ASSISTANT TURN 1
 Let's work through this logic puzzle step by step to figure out which item is in each box.

<think>
 First, I need to list the boxes and their constraints clearly.
 </think>

We have:

- **Red box**
- **Blue box**
- **Green box**

Each contains one of:

- **Coin**
- **Key**
- **Ring**

Constraints:

1. **Red box does not contain the coin**
2. **Blue box contains neither the ring nor the key**
3. **Green box does not contain the ring**

Let's write these out more clearly for analysis.

<think>
 Start with constraint 2: Blue box cannot have ring or key. That leaves only the coin for blue.
 </think>

From constraint 2, the only item that can be in the blue box is the **coin**.

<think>
 Now that blue has the coin, we can eliminate coin from other boxes.
 </think>

So:

- **Blue box:** Coin
- **Red box:** Not coin (so must be key or ring)
- **Green box:** Not ring (so must be coin or key)

Since blue has the coin, red and green can't have the coin anymore. That means:

- **Red box:** Key or ring
- **Green box:** Key (since coin is already taken by blue)

So we now know:

- **Blue box:** Coin
- **Green box:** Key
- **Red box:** Must be ring (only item left)

<think>
 Let's check all constraints to be sure.
 </think>

Final assignments:

- **Red box:** Ring ✓ (satisfies constraint 1)
- **Blue box:** Coin ✓ (satisfies constraint 2)
- **Green box:** Key ✓ (satisfies constraint 3)

All constraints are satisfied.

<think>
 I can now summarize this clearly for the user.
 </think>

Answer:

- **Red box contains the ring.**
- **Blue box contains the coin.**
- **Green box contains the key.**

TIME-32B conversation (out of distribution scenario without timestamp)

929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964

C Training Details

This appendix describes the concrete training pipeline used to obtain all TIME checkpoints. Our goal was to induce a new reasoning policy under realistic hardware constraints rather than to retrain full models. All experiments were therefore designed to be reproducible on a single workstation and to reuse the same optimization scaffold across phases whenever possible, so that differences in behavior can be attributed to data and curriculum rather than to hidden hyperparameter drift.

For each Qwen3 size, we keep the backbone weights frozen and train lightweight adapters with QLoRA. The same adapter configuration is used for all curriculum phases, and adapters are reinitialized when moving from one phase to the next. Phases 1 to 3 share an identical optimization setup and differ only in data distribution and curriculum structure. Phase 4 keeps the same adapter placement but switches to a full-batch regime over a small, high entropy alignment set. This separation lets us isolate which changes are driven by structural seeding, temporal exposure, and contextual modulation, and which are due to the final convergence step.

We also keep decoding and quantisation choices aligned with realistic deployment conditions. Training uses 4 bit quantisation with low rank adapters and gradient checkpointing to fit 32B backbones comfortably on a single high memory GPU. Inference for evaluation uses uniform decoding parameters across all models, and separate ablations compare quantised and full precision runs. Together, these constraints make the reported setting reproducible and close to what a practitioner could implement without large scale infrastructure.

Model and optimization details common to all phases 965 966

- LoRA based fine tuning on a pretrained LLM using **4 bit QLoRA** with **LoRA rank 32**, alpha 32, dropout 0.05. 967 968 969
- Target modules included attention and MLP projection layers (q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj). 970 971 972
- Gradient checkpointing enabled. 973
- Full model not fine tuned (PEFT setup), only adapter weights are updated. 974 975

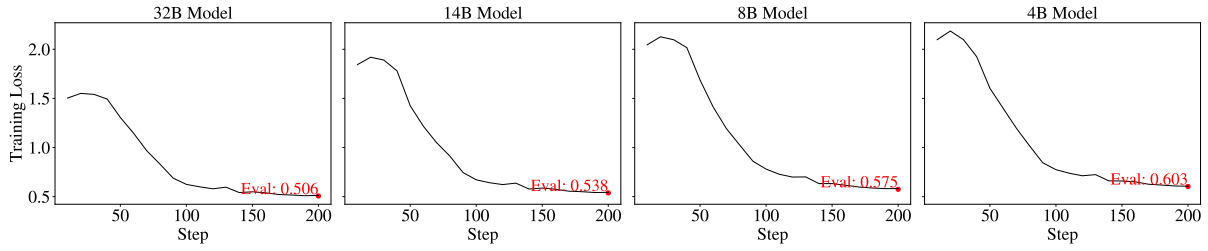


Figure 4: Training Loss Curve and Evaluation Loss for Phase 1

<p>976 Phase 1 – Training Configuration</p> <p>977 Dataset Size</p> <ul style="list-style-type: none"> 978 • Train set: 2,188 samples 979 • Test set: 387 samples <p>980 Sequence Statistics</p> <ul style="list-style-type: none"> 981 • Train sequences: <ul style="list-style-type: none"> 982 – Max length: 2,057 tokens 983 – Mean: 310.8 tokens 984 – 90th percentile: 527 tokens 985 • Test sequences: <ul style="list-style-type: none"> 986 – Max length: 1,504 tokens 987 – Mean: 311.1 tokens 988 – 90th percentile: 544 tokens 	<p>Training Configuration</p> <ul style="list-style-type: none"> • Optimizer: adamw_8bit • Effective batch size: 32 (8 × 4 gradient accumulation) • Epochs: 3 • Learning rate: 2e-5 (linear scheduler with 100 warmup steps) • Max gradient norm: 1.0 • Weight decay: 0.01 • Logging every 10 steps • Evaluation used the held-out test set (387 samples) • Fixed random seed for reproducibility 	<p>989</p> <p>990</p> <p>991</p> <p>992</p> <p>993</p> <p>994</p> <p>995</p> <p>996</p> <p>997</p> <p>998</p> <p>999</p> <p>1000</p> <p>1001</p>
---	---	--

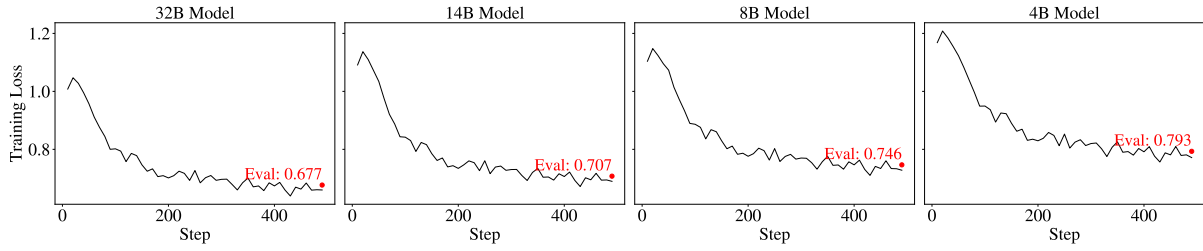


Figure 5: Training Loss Curve and Evaluation Loss for Phase 2

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

Training Configuration

- Optimizer: adamw_8bit
- Effective batch size: 32 (8×4 gradient accumulation)
- Epochs: 3
- Learning rate: $2e-5$ (linear scheduler with 100 warmup steps)
- Max gradient norm: 1.0
- Weight decay: 0.01
- Logging every 10 steps
- Evaluation performed on the updated 935-sample test set
- Fixed random seed for reproducibility

1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030

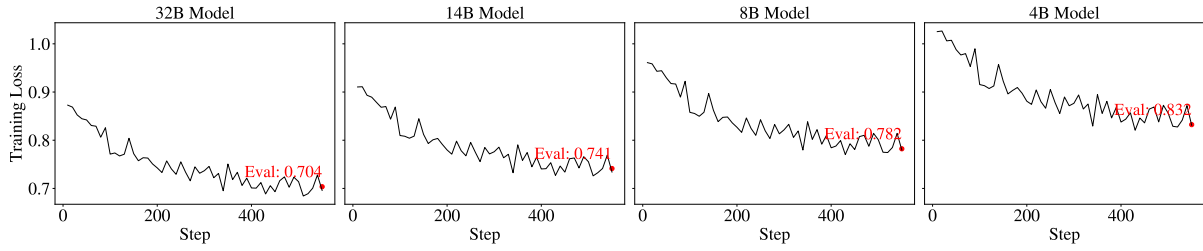


Figure 6: Training Loss Curve and Evaluation Loss for Phase 3

1031	Phase 3 – Training Configuration	Training Configuration	1047
1032	Dataset Size	• Optimizer: adamw_8bit	1048
1033	• Initial Train set: 4,147 samples	• Effective batch size: 32 (8 × 4 gradient accu-	1049
1034	• Initial Test set: 732 samples	mulation)	1050
1035	• After replay integration:	• Epochs: 3	1051
1036	– Final Train set: 5,878 samples	• Learning rate: 2e-5 (linear scheduler with 100	1052
1037	– Final Test set: 1,039 samples	warmup steps)	1053
1038	Sequence Statistics	• Max gradient norm: 1.0	1054
1039	• Train sequences:	• Weight decay: 0.01	1055
1040	– Max length: 3,795 tokens	• Logging every 10 steps	1056
1041	– Mean: 496.0 tokens	• Evaluation performed on 1,039-sample test	1057
1042	– 90th percentile: 855 tokens	set after replay augmentation	1058
1043	• Test sequences:	• Fixed random seed for reproducibility	1059
1044	– Max length: 3,041 tokens		
1045	– Mean: 464.7 tokens		
1046	– 90th percentile: 752 tokens		

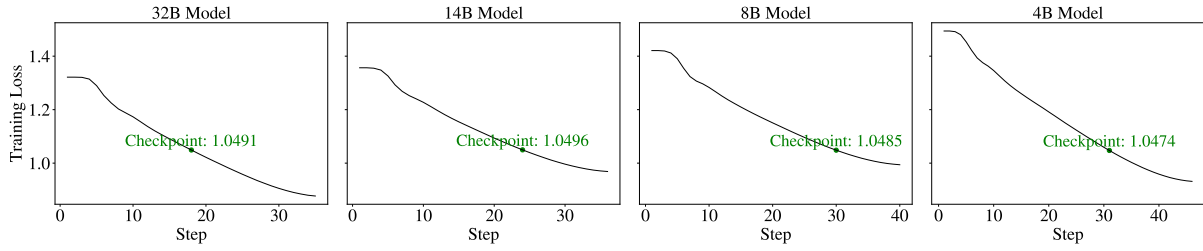


Figure 7: Training Loss Curve and Selected Checkpoints for Phase 4

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 (8×16 gradient accumulation)
- **Learning rate:** $1.5e-4$
- **Max steps** (model-dependent):
32B: 35 14B: 36 8B: 40 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 18 (1.0491)
- **14B:** Step 24 (1.0496)
- **8B:** Step 30 (1.0485)
- **4B:** Step 31 (1.0474)

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097	D Evaluation Methodology		
1098	TIMEBench is designed as a general benchmark		
1099	of temporal intelligence applicable to <i>any</i> lan-		
1100	guage model. Its format imposes no assumptions		
1101	about model-specific tokenization, system instruc-		
1102	tion templates, or internal mechanisms such as tick		
1103	events or transient <think> blocks. Even in multi-		
1104	turn scenarios, no structural markers unique to		
1105	TIME are present—allowing evaluation of general-		
1106	purpose models under standardized temporal rea-		
1107	soning conditions.		
1108	Scenario Generation and Sampling Seeds		
1109	• Master Seed: 3407		
1110	• RNG Engine: NumPy’s PCG64 implementa-		
1111	tion to generate 770 seeds		
1112	• Scenarios: 77 hand-crafted base instances (11		
1113	per diagnostic category). Check Appendix E		
1114	for instances from each category.		
1115	• Variations: Each scenario sampled 10× via		
1116	seed permutations → 770 evaluation runs per		
1117	model		
1118	Decoding Configuration		
1119	All model evaluations use identical sampling pa-		
1120	rameters to ensure comparability:		
1121	• Temperature: 0.6		
1122	• Top-p (nucleus sampling): 0.95		
1123	• Top-k: 20		
1124	• Min-p: 0		
1125	This configuration follows best practices for gen-		
1126	erative control in thinking mode as reported by		
1127	Qwen3 Technical Report (Qwen-Team, 2025).		
1128	Judge Model Protocol		
1129	• Judge Model: gpt-5.2-2025-12-11 (Ope-		
1130	nAI, snapshot dated December 12, 2025)		
1131	• Temperature: 0.0		
1132	• Input:		
1133	– The model’s generated output		
1134	– A structured objective describing the tar-		
1135	get success behavior		
1136	• Blind Evaluation: The judge has no access		
1137	to:		
		– Original scenario	1138
		– Turn timestamps	1139
		– Dialogue context or history	1140
		– Prompt formatting or system instructions	1141
		This isolates behavioral performance from	1142
		surface-level biases or format familiarity.	1143
		Scoring and Aggregation	1144
		• Run-Level: Each output scored binary (pass	1145
		= 1, fail = 0)	1146
		• Scenario-Level: Mean of 10 run scores per	1147
		scenario	1148
		• Category-Level: Mean of 11 scenario scores	1149
		expressed as percentage	1150
		• Benchmark Score: Mean of 7 category	1151
		scores	1152
		Confidence Interval Estimation	1153
		We compute 95% confidence intervals using strat-	1154
		ified bootstrapping (10,000 replicates):	1155
		1. Resample scenario-level scores <i>within each</i>	1156
		<i>category</i>	1157
		2. Recompute category and overall means	1158
		3. Take 2.5th to 97.5th percentile range as the	1159
		confidence interval	1160
		This method captures scenario-level uncertainty	1161
		without conflating seed-level noise.	1162

1163	Structural and Reasoning Format Analysis	Aggregation Strategy	1200
1164	We conducted a detailed analysis of structural, formatting, and reasoning behaviors across all 770	• All stats are computed on individual runs (n = 770)	1201
1165	TIMEBench outputs per model. This posthoc pass	• These are then aggregated per scenario (mean across the 10 runs)	1202
1166	provides a high-resolution lens on how <think>	• This yields 77 scenario-level aggregates per model	1203
1167	blocks and markdown formatting are used in practice—beyond simple accuracy metrics.	• 95% confidence intervals are calculated using bootstrap resampling (n = 10,000) at the scenario level to avoid seed-level noise	1204
1168			1205
1169			1206
1170	Objective		1207
1171	To evaluate:		1208
1172	• Reasoning placement: Where in the output <think> blocks appear		1209
1173		Tokenizer	1210
1174	• Reasoning scale: Number and token count of reasoning blocks	• Tokenization used Qwen/Qwen3-32B tokenizer from Hugging Face for all measurements to align with training/inference setup	1211
1175			1212
1176	• Formatting dynamics: Use of light markdown (bold/italics) and heavy markdown (lists, headers, tables, etc)		1213
1177		Outputs	1214
1178		We compute:	1215
1179	• Structural failures: Malformed or degenerate outputs (e.g., unmatched tags, format bleed, infinite repetitions)	• Descriptive stats: Means as minimum values, medians, maximum values over all 770 runs	1216
1180			1217
1181		• Bootstrapped stats: Scenario-level confidence intervals per metric	1218
1182	• Style evolution: Changes in formatting complexity, redundancy, and context-dependence		1219
1183		These include:	1220
1184	Annotation Pipeline	• Mean number of <think> blocks per response	1221
1185	Each run was annotated using a tokenizer-aligned pattern-matching pipeline, which extracts:	• Mean number of tokens used for thinking per response	1222
1186			1223
1187	• Output length and reasoning token stats	• Markdown usage rates (light vs heavy) per scenario	1224
1188			1225
1189	• Markdown usage and breakdown into light/heavy categories	• Percentage of degenerate outputs	1226
1190	• <think> block count, token distribution, and position (start, middle, end)	• Think block positional distribution (start/middle/end)	1227
1191			1228
1192	• Detection of malformed structures (e.g., unbalanced tags)	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.	1229
1193			1230
1194	• Detection of infinite repetitions (e.g., persistent looping outputs)		1231
1195			1232
1196	• Reasoning leakage (e.g., reflective text outside <think>)		1233
1197			1234
1198	• Markdown artifact bleed (e.g., tables inside <think>)		1235
1199			

1236

E TIMEBench Completion Examples

1237

Category: Chronological Retrospection

1238

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>).

1239

1240

1241

1242

1243

The correct answer is either:

1244

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

1245

1246

- **1 year, 3 months, 19 days** (accept 1 year, 3 months, 18 days; minor rounding/interpretation of start/end day is allowed)

1247

1248

1249

- **15 months, 19 days** (accept 15 months, 18 days; minor rounding/interpretation of start/end day is allowed)

1250

1251

1252

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", "over 15 months", etc fail the test as well.

1253

1254

1255

1256

1257

Model: TIME-32B

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>2022-12-10T18:05:31</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.

1258 **Category: Invalid Time Detection**

1259 **Objective:** The assistant must notice that the
1260 user's message has a timestamp of February 30,
1261 2028—an impossible date, as February never has
1262 30 days. The assistant must note this anomaly
1263 either in the answer or in a think block. Providing
1264 the interpolation pipeline is acceptable, but there
1265 must be some acknowledgment of the impossible
1266 date.

1267 **Model: TIME-32B**

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.

1268 **Category: Temporal Adaptivity**

1269 **Objective:** The assistant must demonstrate temporal adaptivity by showing, in any way, that it recognizes the user’s deadline (5:00 PM) is only minutes away at the time of the final query (4:57 PM). Any sign of urgency, time sensitivity, or reference to the approaching deadline—in the response or in a think block—satisfies the requirement. Failing to notice the time constraint fails the test.

1270 **Model: TIME-14B**

1271
1272
1273
1274
1275
1276
1277

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!

1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288

Category: Temporal Flow Anomaly Detection

Objective: The assistant must notice the massive timeskip (over a hundred years) between the user's messages and recognize it as an anomaly. This can be expressed either in the user-facing text or in a think block (e.g., flagging the unusual gap or questioning the date or believing something is amiss). Detecting the gap as anomalous or highly irregular is required even if the model then chooses to continue assisting as requested.

Model: TIME-14B

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]

1289 **Category: Timezone Sensitivity**

1290 **Objective:** The assistant must demonstrate time-
1291 zone sensitivity by recognizing that the second user
1292 message corresponds to a local time of -06:00 and
1293 the others are from San Francisco (-08:00) and
1294 Rome (+01:00). When asked about the layover lo-
1295 cation, the assistant should use the timezone offset
1296 (-06:00) and time of the second turn to reason and
1297 correctly infer a likely city such as Chicago. Fail-
1298 ing to use the timezone clues to deduce the layover
1299 location fails the test.

1300 **Model: TIME-8B**

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!

1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311

Category: Temporal Contextual Awareness

Objective: The assistant must demonstrate temporal contextual awareness by showing, in any way, recognition that the user is asking this question in context of Christmas, that the reason for gift buying and preparation is for Christmas. Any acknowledgment (explicit or subtle, in the answer or a think block) of the holiday context is sufficient. Failing to notice the special temporal context and responding generically fails the test.

Model: TIME-8B

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!

1312 **Category: Time Gap Awareness**

1313 **Objective:** The assistant must demonstrate time
1314 gap awareness—a recognition that time has passed
1315 and the user’s situation may have evolved since
1316 their initial inquiry. The assistant should avoid
1317 assuming nothing has changed, and offer context-
1318 appropriate advice or clarification. Any evidence of
1319 implicit or explicit recognition or the fact that the
1320 model is aware of the time gap is enough. Failing
1321 to show this awareness fails the test.

1322 **Model: TIME-4B**

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 Alacritty 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?

1323 **Category: Temporal Adaptivity**

1324 **Objective:** The assistant must demonstrate temporal adaptivity by recognizing that more than six hours have passed since the user went into the 7:00 PM showing of *Spider-Man: No Way Home*. The model should show awareness that the user has now likely seen the film, and adapt its response accordingly (e.g., being open to discussing feats from *No Way Home*). Any acknowledgment of the time elapsed or the user's likely experience in the response or think block is sufficient. Failing to notice that the user has seen the movie fails the test.

1330 **Model: TIME-4B**

1332 **USER TURN 1**

1333 <time>2023-12-22T18:53:00</time>

1334 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!

1335 **ASSISTANT TURN 1**

1336 <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.

1337 **USER TURN 2**

1338 <time>2023-12-23T00:59:00</time>

1339 Okay, so feat-wise, which Spider-Man of the three do you think is the strongest in the movies?

1340 **ASSISTANT TURN 2**

1341 <think>

1342 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 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.

F Detailed Ablations and Metrics

This section presents a phase-wise and model-wise ablation of structural, formatting, and behavioral statistics for Qwen3 baselines (thinking and no-thinking) and TIME models. 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 thinking Modes 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 thinking token distributions**
- **Think block frequency and placement**
- **Markdown usage (light/heavy):** light (***bold/italics***/heavy markdown(lists, headers, tables, etc)
- **Degenerate output 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 F.1.1: Structural and Behavioral Metrics (4B Models)

Statistic	Qwen3 4B		TIME-4B	WSR p
	No Think	Think		
Benchmark Score	17.53 (11.95–23.38)	30.13 (23.90–36.36)	52.60 (44.55–60.39)	3.8e–04
Chronological Retrospection	30.00 (10.00–52.73)	53.64 (31.82–75.45)	42.73 (23.64–62.73)	—
Invalid Time Detection	6.36 (0.00–19.09)	20.91 (1.82–42.73)	43.64 (20.00–67.27)	—
Temporal Adaptivity	5.45 (0.00–12.73)	13.64 (0.00–28.18)	49.09 (27.27–70.91)	—
Temporal Contextual Awareness	32.73 (8.18–59.09)	40.00 (16.36–64.55)	76.36 (56.36–92.73)	—
Temporal Flow Anomaly Detection	0.00 (0.00–0.00)	1.82 (0.00–4.55)	40.91 (19.09–63.64)	—
Time Gap Awareness	0.00 (0.00–0.00)	2.73 (0.00–8.18)	57.27 (35.45–78.18)	—
Timezone Sensitivity	48.18 (30.00–66.36)	78.18 (63.64–90.91)	58.18 (41.82–73.64)	—
Mean Output Tokens / Run	621.01 (543.56–700.11)	1753.13 (1477.36–2062.26)	369.28 (318.18–424.01)	<1e–08
Mean Thinking Tokens / Run	0.00 (0.00–0.00)	1116.37 (877.99–1392.25)	85.56 (68.29–110.78)	<1e–08
Mean # Think Blocks / Run	0.00 (0.00–0.00)	0.98 (0.96–1.00)	2.01 (1.23–3.41)	4.8e–05
% Runs w/ Think Blocks	0.00 (0.00–0.00)	98.31 (96.49–99.61)	87.40 (81.69–92.47)	—
% Heavy Markdown	87.79 (81.82–92.99)	85.97 (78.96–92.08)	68.83 (60.00–77.40)	—
% Light Markdown	95.58 (91.43–98.83)	95.71 (91.56–98.96)	90.91 (86.62–94.68)	—
% Any Degeneracy	3.38 (1.43–5.84)	20.13 (13.77–26.88)	5.19 (2.47–8.96)	2.5e–05
% Malformed Outputs	0.00 (0.00–0.00)	1.69 (0.39–3.51)	1.30 (0.52–2.21)	—
% Infinite Repetitions	2.47 (1.17–4.29)	1.69 (0.39–3.51)	0.13 (0.00–0.39)	—
% Reasoning Leakage	0.91 (0.00–2.60)	3.38 (1.69–5.45)	1.17 (0.26–2.47)	—
% Formatting Leakage	0.00 (0.00–0.00)	16.88 (10.91–23.64)	3.25 (0.91–6.49)	—

Note. WSR = Wilcoxon signed-rank test, computed between Qwen3 thinking mode and TIME across 77 scenario-level comparisons.

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

Statistic	Qwen3 8B		TIME-8B	WSR p
	No Think	Think		
Benchmark Score	21.56 (15.32–28.05)	32.99 (26.88–39.09)	59.87 (53.38–66.23)	1.9e–05
Chronological Retrospection	24.55 (8.18–43.64)	58.18 (36.36–78.18)	50.00 (33.64–65.45)	—
Invalid Time Detection	7.27 (0.00–20.91)	18.18 (3.64–37.27)	45.45 (23.64–67.27)	—
Temporal Adaptivity	21.82 (1.82–44.55)	20.00 (4.55–38.18)	80.00 (67.27–90.91)	—
Temporal Contextual Awareness	34.55 (8.18–61.82)	41.82 (16.36–69.09)	70.00 (50.00–87.27)	—
Temporal Flow Anomaly Detection	0.00 (0.00–0.00)	1.82 (0.00–4.55)	44.55 (30.91–58.18)	—
Time Gap Awareness	0.00 (0.00–0.00)	0.91 (0.00–2.73)	64.55 (48.18–79.09)	—
Timezone Sensitivity	62.73 (41.82–81.82)	90.00 (81.82–97.27)	64.55 (45.45–81.82)	—
Mean Output Tokens / Run	607.29 (528.02–690.00)	1674.88 (1389.19–2001.83)	351.63 (312.57–392.46)	<1e–08
Mean Thinking Tokens / Run	0.00 (0.00–0.00)	1025.63 (776.60–1317.86)	87.31 (72.62–104.71)	<1e–08
Mean # Think Blocks / Run	0.00 (0.00–0.00)	0.98 (0.96–0.99)	2.06 (1.64–2.62)	6.7e–08
% Runs w/ Think Blocks	0.00 (0.00–0.00)	97.92 (95.84–99.35)	86.23 (80.39–91.56)	—
% Heavy Markdown	80.78 (73.25–87.66)	84.94 (77.79–91.30)	66.23 (57.53–74.81)	—
% Light Markdown	94.55 (90.00–98.18)	95.45 (91.56–98.44)	86.88 (81.17–91.95)	—
% Any Degeneracy	5.19 (2.73–8.31)	17.92 (11.56–24.68)	5.58 (2.60–9.48)	1.8e–03
% Malformed Outputs	0.00 (0.00–0.00)	2.08 (0.65–4.16)	2.21 (0.91–3.90)	—
% Infinite Repetitions	4.81 (2.46–7.79)	2.08 (0.65–4.16)	0.26 (0.00–0.78)	—
% Reasoning Leakage	0.39 (0.00–1.17)	2.99 (1.43–4.94)	0.91 (0.13–2.21)	—
% Formatting Leakage	0.00 (0.00–0.00)	15.06 (9.09–21.69)	2.34 (0.52–4.55)	—

Note. WSR = Wilcoxon signed-rank test, computed between Qwen3 thinking mode and TIME across 77 scenario-level comparisons.

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

Statistic	Qwen3 14B		TIME-14B	WSR p
	No Think	Think		
Benchmark Score	29.48 (22.47–36.36)	34.42 (28.44–40.65)	64.80 (59.09–70.39)	1.6e–06
Chronological Retrospection	64.55 (38.18–88.18)	55.45 (34.52–76.36)	55.45 (39.09–70.00)	—
Invalid Time Detection	20.91 (0.00–43.64)	19.09 (1.82–41.82)	41.82 (25.45–59.09)	—
Temporal Adaptivity	18.18 (2.73–36.36)	23.64 (9.09–40.91)	84.55 (76.36–91.82)	—
Temporal Contextual Awareness	32.73 (8.18–59.09)	42.73 (18.18–69.09)	78.18 (63.64–90.00)	—
Temporal Flow Anomaly Detection	0.00 (0.00–0.00)	1.82 (0.00–4.55)	66.36 (48.18–83.64)	—
Time Gap Awareness	0.00 (0.00–0.00)	4.55 (0.00–10.00)	61.82 (43.64–79.09)	—
Timezone Sensitivity	70.00 (49.09–87.27)	93.64 (90.00–97.27)	65.45 (50.91–80.00)	—
Mean Output Tokens / Run	563.03 (486.58–641.05)	1514.66 (1270.89–1785.34)	317.21 (283.39–353.33)	<1e–08
Mean Thinking Tokens / Run	0.00 (0.00–0.00)	887.42 (683.44–1127.52)	91.19 (76.67–111.07)	<1e–08
Mean # Think Blocks / Run	0.00 (0.00–0.00)	0.99 (0.98–1.00)	1.90 (1.48–2.55)	1.3e–08
% Runs w/ Think Blocks	0.00 (0.00–0.00)	99.09 (98.05–99.87)	91.17 (86.36–95.45)	—
% Heavy Markdown	83.51 (75.84–90.26)	86.36 (79.48–92.34)	61.82 (52.60–70.78)	—
% Light Markdown	95.58 (90.78–99.35)	97.40 (94.16–99.61)	86.75 (81.43–91.43)	—
% Any Degeneracy	4.16 (1.56–7.66)	15.45 (9.09–22.21)	2.08 (0.65–3.77)	5.3e–05
% Malformed Outputs	0.00 (0.00–0.00)	0.91 (0.13–2.08)	0.26 (0.00–0.65)	—
% Infinite Repetitions	4.16 (1.69–7.66)	1.30 (0.13–2.99)	0.00 (0.00–0.00)	—
% Reasoning Leakage	0.00 (0.00–0.00)	1.30 (0.39–2.47)	0.78 (0.00–1.82)	—
% Formatting Leakage	0.00 (0.00–0.00)	14.29 (8.18–21.04)	1.17 (0.13–2.47)	—

Note. WSR = Wilcoxon signed-rank test, computed between Qwen3 thinking mode and TIME across 77 scenario-level comparisons.

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

Statistic	Qwen3 32B		TIME-32B	WSR p
	No Think	Think		
Benchmark Score	31.82 (25.71–38.31)	37.40 (31.56–43.51)	64.81 (58.18–71.17)	5.0e–07
Chronological Retrospection	61.82 (38.18–83.64)	60.00 (41.82–77.27)	63.64 (45.45–80.00)	—
Invalid Time Detection	11.82 (2.73–25.45)	31.82 (13.64–51.82)	52.73 (32.73–71.82)	—
Temporal Adaptivity	20.00 (1.82–44.55)	26.36 (7.27–47.27)	76.36 (64.55–87.27)	—
Temporal Contextual Awareness	40.00 (13.64–67.27)	43.64 (20.00–69.09)	76.36 (58.18–92.73)	—
Temporal Flow Anomaly Detection	0.91 (0.00–2.73)	3.64 (0.00–8.18)	44.55 (23.64–65.45)	—
Time Gap Awareness	0.91 (0.00–2.73)	3.64 (0.00–7.27)	58.18 (40.00–75.45)	—
Timezone Sensitivity	87.27 (78.18–94.55)	92.73 (83.64–99.09)	81.82 (68.18–92.73)	—
Mean Output Tokens / Run	608.96 (533.37–686.50)	1573.47 (1327.94–1856.57)	332.64 (296.15–371.77)	<1e–08
Mean Thinking Tokens / Run	0.00 (0.00–0.00)	910.52 (705.34–1158.80)	84.16 (71.18–98.27)	<1e–08
Mean # Think Blocks / Run	0.00 (0.00–0.00)	0.99 (0.98–1.00)	1.67 (1.34–2.06)	1.4e–04
% Runs w/ Think Blocks	0.00 (0.00–0.00)	99.22 (97.92–100.00)	80.65 (73.51–87.40)	—
% Heavy Markdown	83.64 (77.01–89.61)	90.91 (85.19–95.71)	62.73 (53.51–71.69)	—
% Light Markdown	95.84 (92.60–98.31)	98.31 (95.45–99.87)	85.45 (79.74–90.65)	—
% Any Degeneracy	4.42 (2.60–6.49)	18.18 (11.43–25.32)	3.64 (1.95–5.97)	4.2e–04
% Malformed Outputs	0.00 (0.00–0.00)	0.78 (0.00–2.08)	1.95 (1.04–2.99)	—
% Infinite Repetitions	4.29 (2.47–6.23)	0.91 (0.00–2.47)	1.30 (0.52–2.34)	—
% Reasoning Leakage	0.13 (0.00–0.39)	0.91 (0.13–1.95)	1.17 (0.52–1.95)	—
% Formatting Leakage	0.00 (0.00–0.00)	17.01 (10.39–24.16)	1.17 (0.13–2.86)	—

Note. WSR = Wilcoxon signed-rank test, computed between Qwen3 thinking mode and TIME across 77 scenario-level comparisons.

Appendix F.2: Phase-wise Ablation Details (32B Models)

Statistic	Qwen3 32B		Phase 1	Phase 2	Phase 3	TIME (32B)
	No-Thinking	Thinking				
Benchmark Score	31.82	37.40	42.47	56.88	52.08	64.81
Chronological Retrospection	61.82	60.00	63.64	52.73	65.45	63.64
Invalid Time Detection	11.82	31.82	39.09	21.82	30.91	52.73
Temporal Adaptivity	20.00	26.36	40.91	80.00	68.18	76.36
Temporal Contextual Awareness	40.00	43.64	48.18	50.00	49.09	76.36
Temporal Flow Anomaly Detection	0.91	3.64	7.27	49.09	40.00	44.55
Time Gap Awareness	0.91	3.64	3.64	59.09	44.55	58.18
Timezone Sensitivity	87.27	92.73	94.55	85.45	66.36	81.82
Mean Output Tokens / Run	608.96	1573.47	1434.56	362.45	294.51	332.64
Mean Thinking Tokens / Run	0.00	910.52	803.52	76.59	52.94	84.16
Mean # Think Blocks / Run	0.00	0.99	0.99	1.12	1.25	1.67
% Runs w/ Think Blocks	0.00	99.22	99.48	95.58	89.22	80.65
% Heavy Markdown	83.64	90.91	88.57	65.32	61.43	62.73
% Light Markdown	95.84	98.31	97.92	83.38	79.48	85.45
% Any Degeneracy	4.42	18.18	13.90	4.68	0.78	3.64
% Malformed Outputs	0.00	0.78	0.52	1.82	0.26	1.95
% Infinite Repetitions	4.29	0.91	0.52	0.39	0.13	1.30
% Reasoning Leakage	0.13	0.91	1.04	2.47	0.26	1.17
% Formatting Leakage	0.00	17.01	12.47	1.17	0.26	1.17