STATETRACKING: BENCHMARKING COMPLETE CURRENT-STATE RECONSTRUCTION FROM EVOLVING HISTORIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models increasingly serve as long-running assistants in settings such as AI-assisted coding and workflow management, where users expect them to maintain an accurate account of the current state despite long histories of edits, deletions, and updates. Existing benchmarks study long-term memory, symbolic state changes, and file reconstruction from revision histories, but often rely on query-specific evaluation, synthetic data, or real-world histories without explicit factor control, leaving them unable to jointly test evolving text collection recovery and complete final values for all active items (called keys). This task probes whether models can selectively update information associated with each key while preserving valid content and excluding information invalidated by later changes. To fill this evaluation gap, we introduce StateTracking, a benchmark for complete current-state reconstruction from an initial state and an ordered history. It contains 3,420 histories adapted from five datasets spanning document editing, online discussions, software issue workflows, financial orders, and network records, with workload difficulty characterized by three factors: final active-key count N, transition count T, and deletion frequency ρ. Three datasets–Wikipedia Article, Wikipedia Talk, and Jira–are grounded in real revision or workflow histories. The other two—LOBSTER and DNS—provide controlled workloads with broader coverage of these factors, enabling systematic evaluation across (N, T, ρ) conditions and scalability analysis as state size and history length increase. Completed evaluations reveal a gap between recovering the final keys and their complete values. Error analysis also identifies deleted keys and superseded text in model outputs, alongside distinct text-fidelity and formatting failures.

1 INTRODUCTION

Reliable assistance over extended interactions requires an accurate account of what remains valid in the current state. In long-running settings such as AI-assisted coding, collaborative editing, and workflow management, users may repeatedly create, modify, complete, or remove tasks, documents, issues, and records. As these changes accumulate, the interaction history contains both information that is still relevant and earlier states that are no longer valid. We represent each tracked item by a key, which identifies it in the state, and a value containing its text or fields. A useful assistant must therefore do more than remember individual updates: it must determine which keys remain active and recover their complete values at the end of the history.

Tracking this evolving state is challenging because updates are typically local while the required answer is global. A new message may modify only the value associated with one key, remove an active key, or leave most of the previous state untouched. Recovering the final state therefore requires integrating changes across a potentially long history, discarding invalidated information, and preserving content that is never mentioned again after its last valid update.

Existing benchmarks evaluate complementary aspects of evolving state. Entity Tracking (Kim & Schuster, 2023) studies sequential updates to symbolic entities and collections, including complete symbolic-state prediction, but focuses on relatively short, structured update sequences rather than reconstructing complete states after extended real-world histories. Memora (Uddin et al., 2026) eval-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

uates the persistence, mutation, and invalidation of information over long user–assistant interactions, including whether updated or removed information is handled correctly. However, its evaluation primarily targets individual facts or collection membership rather than complete values for all active keys. LoCoDiff (Mentat AI Team, 2025) reconstructs the complete text of a file from real revision histories. Because the target file is specified in advance, the model reconstructs its contents without identifying which files remain in the repository.

While these benchmarks cover important components of state tracking, including sequential updates, invalidation, and complete text reconstruction, they fail to jointly test the identification of active keys in a real text collection and the reconstruction of all their final values. We therefore ask: can a model reconstruct the entire state that is valid now, without being told which keys remain active or which key to query? In long-running assistant workflows, this means identifying the active keys for tasks, documents, or records and recovering their complete final values. At the model level, this tests whether later changes selectively override earlier information without disrupting content that remains valid. Reconstructing the complete state makes failures of updating, preservation, and invalidation observable across all keys in the same answer.

We call this problem *complete current-state reconstruction*: given an initial state and an ordered history of changes, recover all keys active at the end of that history and their complete final values. Here, current refers to the end of the supplied history. The task requires updating the values of retained keys, excluding removed keys, and preserving unchanged information that remains valid.

To understand what makes complete-state reconstruction difficult, we consider three workload factors. Two are natural properties of an evolving state: the final active-key count N determines how many keys must be returned, while the transition count T determines how many changes must be integrated to reach it. Deletion introduces a qualitatively different requirement. In addition to incorporating updates, a model must prevent previously valid keys or text from appearing in the final state once they have been deleted. Motivated by this distinction, we introduce deletion frequency ρ to measure the fraction of changes involving deletions of text, sections, or records. Together, N, T, and ρ provide a complementary view of reconstruction difficulty by capturing distinct but interacting workload dimensions. Their joint characterization is important because similar final states can result from markedly different update and deletion histories, which may impose very different reconstruction demands.

To systematically evaluate complete current-state reconstruction along these factors, we introduce **StateTracking**, comprising 3,420 histories adapted from five datasets spanning document editing, online discussions, software issue workflows, financial orders, and network records. The three realistic datasets—Wikipedia Article, Wikipedia Talk, and Jira—are organized into 27 predefined (N, T, ρ) bins, with 20 histories per bin. Wikipedia Article and Wikipedia Talk are constructed from real revision histories of encyclopedia articles and discussion pages, while Jira is primarily based on real issue histories, with synthetic histories used only to fill bins that are sparse in the observed corpus. LOBSTER and DNS provide fully synthetic workloads on a prescribed $N \times T \times \rho$ grid. This design allows us to compare reconstruction performance across controlled workload conditions within each domain and examine how the resulting errors differ.

Figure 1 makes the reconstruction requirement concrete: a removed Wikipedia section disappears, an edited section incorporates new lines, and an untouched section remains complete. Wikipedia provides comparisons across a balanced grid for reconstructing section texts, while Jira, LOBSTER, and DNS extend the same complete-state task to structured values. Interpreting these comparisons additionally accounts for text length, source dependence, and the operation mix associated with each setting.

The evaluations reveal failures in both key recovery and value reconstruction. Across the four models evaluated on all 540 Wikipedia Talk histories, 66.1–77.8% of answers recover every final key but contain inexact text. History inspection identifies returned deleted sections and obsolete versions of surviving content. Other failures involve whitespace, output formatting, or exhausted generation budgets. The reappearance of deleted sections and superseded passages shows that models can reproduce historical content without consistently enforcing later edits and deletions. Such failures to update or invalidate content are distinct from the copying and formatting errors also counted by exact match.

Our main contributions are:

- A benchmark adapted from five datasets that unifies complete current-state reconstruction across realistic sources (Wikipedia Article, Wikipedia Talk, and Jira) and controlled synthetic workloads (LOBSTER and DNS), with explicit control over update characteristics (N, T, and ρ)—spanning diverse update semantics from long-form free text to short structured fields.
- Diagnostics that distinguish membership errors and outdated values from text-fidelity and output failures, exposing cases where an incorrect answer retains information invalidated by later changes.

The dataset is included in the supplementary materials, and we will publicly release it together with reusable construction code upon acceptance of the paper.

```
...
Begin transaction 14.
...
Before key: {"heading":"Academic opinions","page_title":"Theism","
    section_kind":"level2"}
...
Remove the whole section at the before key from the active state.
...
End transaction 14; commit atomically.
...
Begin transaction 32.
...
Before key: {"heading":"See also","page_title":"Theism","section_kind":"
    level2"}
...
Keep 1 context line unchanged:
*[[Atheism]]
Insert 1 new line:
*[[Nontheism]]
...
End transaction 32; commit atomically.
...
Query the complete final active state: all current key/value pairs, including
     full unchanged portions of every retained body.
```

Figure 1: Original prompt excerpts from the Theism Wikipedia page. Pale backgrounds distinguish section removal and text insertion. `...` marks omissions; literal-block delimiters are omitted.

2 Task Definition and Dataset Construction

StateTracking spans three realistic datasets drawn from human-maintained sources—Wikipedia Article, Wikipedia Talk, and Jira—and two controlled synthetic datasets, LOBSTER and DNS. All five require complete current-state reconstruction. Wikipedia values are section texts reconstructed from local edits, whereas Jira, LOBSTER, and DNS values contain structured fields. The realistic datasets bin real revision or issue histories into joint workload conditions; the controlled datasets prescribe those conditions directly and pair trajectories to compare value structures under matched operation sequences. The five datasets contain 3,420 histories in total (Table 1).

2.1 Task formulation

At each step, a state S maps the keys active at that step to their associated values. A task provides an initial state S_0 and an ordered history $H = (C_1, \ldots, C_B)$, where each C_b is a group of changes applied together. Applying the domain's transition rules Φ gives

$$S_b = \Phi(S_{b-1}, C_b), \qquad S^\star = S_B. \tag{1}$$

Table 1: StateTracking contains 3,420 histories across five datasets grouped by source. Keys identify sections, issues, orders, or domain names, and values contain text or record fields. The three realistic datasets provide balanced joint workload bins; the two controlled datasets add prescribed synthetic workloads.

Dataset	History source	Key / value	Size
Realistic histories: 27 joint workload bins per dataset			
Wikipedia Article	Page revisions	Section key / wikitext	540 histories
Wikipedia Talk	Section traces	Section key / plaintext	540 histories
Jira	455 real + 85 synthetic windows	Issue key / three fields	540 windows
Controlled synthetic workloads			
LOBSTER	Synthetic orders	Order ID / three fields	900 trajectories
DNS	Synthetic network records	Domain name / type-specific fields	900 trajectories

Here, S_b denotes the state after applying the first b change groups, S_B is the state obtained after applying all B change groups in the history, and $S^\star$ denotes this reference final state. Let $K^\star = \mathrm{keys}(S^\star)$ be the set of keys that remain active in $S^\star$. The model receives text describing the initial state and history, together with the output instructions, and returns a predicted final state $\hat{S}$. Reconstruction is correct when $\hat{S} = S^\star$: the answer contains exactly the final keys and their complete final values. The final key set $K^\star$ is not supplied to the model.

For illustration, consider Wikipedia, where keys identify sections and values contain their complete final texts. The answer has the form {`"state":[{"key":...,"value":...}]`}. A key for a removed section is omitted. Each retained key requires its section's complete final text, including passages unchanged by the local edits. A key whose section has an empty body remains in the answer with an empty string. Entry order is immaterial, duplicate keys are invalid, and keys and decoded text are compared exactly.

2.2 Workload Coordinates

We describe workloads by final active-key count $N = |K^\star|$, declared transition count $T = \sum_{b=1}^{B} |C_b|$, and deletion frequency $\rho = D/T$, where D is the number of changes involving a deletion. These coordinates distinguish the number of keys to return, the number of recorded changes, and the share of changes involving removal, respectively. For Wikipedia, D counts a change once whenever it removes body text or an entire section. For Jira, LOBSTER, and DNS, D counts changes that remove a key from the tracked state. We use realized N, T, ρ to define joint workload bins and compare model outcomes across their conditions.

2.3 Dataset Construction

2.3.1 Wikipedia Article and Talk

We constructed Wikipedia Article and Talk by collecting recent revisions, assembling section histories, and selecting histories to cover a joint workload grid. We first retrieved page revisions[1] and selected edits by source timestamp to reduce potential training-data leakage. Article used updates dated January 1 through September 9, 2026 (UTC), while Talk used each source page's most recent revisions. Comparing successive versions yielded section additions, removals, renames, and local text changes. We excluded ambiguous section matches so that each retained edit referred to an identifiable section.

We assembled these changes differently for the two datasets. Article retained consecutive revisions of one page and applied all changes within a revision together, preserving dependencies between sections. Talk combined real section histories. Each section started from the state preceding a contiguous sequence of its edits, and we interleaved these sequences without changing their internal order. This composition varied the section count and history length while preserving the observed text and local changes. The input supplied initial bodies and local edits, and the target contained every remaining section with its complete final text.

[1] `https://www.mediawiki.org/wiki/API:Revisions`

Table 2: Realistic workload grids. Each of the three realistic datasets crosses three levels on every axis into 27 joint bins of 20 histories. Wikipedia levels are intervals set by source support; Jira targets discrete (N, T, ρ) values within a tolerance ($\pm 10\%$ in N, T, ± 0.03 in ρ), with synthetic completion of sparse bins.

	Wikipedia Article			Wikipedia Talk			Jira		
Coordinate	Low	Medium	High	Low	Medium	High	Low	Medium	High
N	1–6	7–8	≥ 9	1–4	5–8	≥ 9	64	128	256
T	1–8	9–16	≥ 17	1–64	65–128	≥ 129	256	512	1024
ρ	$[0, .80)$	$[.80, .95)$	$[.95, 1]$	$[0, .50)$	$[.50, .60)$	$[.60, 1]$	0.2	0.3	0.4

We measured each history's N, T, ρ and fixed three intervals per coordinate separately for Article and Talk, choosing boundaries that reduced the number of additional histories needed to fill the grid (Table 2). We selected Article histories and filled sparse Talk conditions with additional order-preserving trace compositions. Each dataset contained 20 histories in all 27 joint bins, yielding 540 histories per dataset. We generated natural-language instructions and a paired formal representation, then verified that public-history replay recovered the reference final state for every history. The 20 histories per bin supported comparisons across workload conditions within each dataset, with text lengths and source-page and section-trace identifiers retained to account for text length and shared source material. Appendix A.1 elaborates dataset construction, and Appendix A.2 gives dataset statistics.

2.3.2 JIRA

We constructed Jira from the public issue histories of Apache projects.[2] Each active key identified a software issue whose value contained priority, assignee, and status. Final statuses (Resolved, Closed, or Done) removed issue keys from the state, reopening could reintroduce the same key, and partial updates inherited unchanged fields. The 455 observed Apache-project windows covered 26 of 27 target bins, with 21 bins meeting the quota of 20 histories. Adding 85 synthetic windows under the same rules brought all 27 bins to that quota without extending any marginal N, T, ρ range. This filled gaps in joint workload coverage while preserving the same ranges along individual axes.

2.3.3 LOBSTER AND DNS

We generated LOBSTER and DNS as fully synthetic datasets on prescribed workload grids. For each dataset, three values of N, three values of T, and five values of ρ formed $3 \times 3 \times 5 = 45$ bins. We generated 20 histories with different seeds per bin, yielding 900 histories per dataset (Table 3). LOBSTER tracked financial orders with side, price, and remaining quantity. DNS tracked domain names with record type, TTL, target, and type-specific fields. Both supplied complete values on insertion and overwrite.

At fixed N and T, varying ρ changed the deletion workload without changing the number of final active keys. Each deletion was paired with a fresh insertion, so higher ρ allocated more changes to deletion and insertion and fewer to overwrites. We matched the two datasets trajectory by trajectory. For each of the 900 bin/seed pairs, LOBSTER and DNS shared the same operation sequence and key-access schedule up to a bijective renaming of keys. The stored values and update rules differed: LOBSTER used three fields for every order, whereas DNS used a type-dependent schema. This pairing held key lifecycles and access patterns fixed while varying value structure. It did not equalize surface token length, so the two datasets were matched in workload but not in input size.

Appendix A.3 elaborates dataset construction for Jira, LOBSTER, and DNS.

3 EXPERIMENTS

Our experiments evaluated complete current-state recovery and its sensitivity to final-state size, history length, and deletion frequency. We then examined failures of updating, preservation, and inval-

[2] https://issues.apache.org/jira/

Table 3: Controlled workload grids. LOBSTER and DNS use the same $3 \times 3 \times 5$ design, with 20 histories per bin (45 bins, 900 histories per dataset).

Dataset	N	T	ρ
LOBSTER	$16, 64, 256$	$256, 1024, 4096$	$0.1, 0.2, 0.3, 0.4, 0.5$
DNS	$16, 64, 256$	$256, 1024, 4096$	$0.1, 0.2, 0.3, 0.4, 0.5$

idation separately from text-fidelity and output errors. The full-dataset evaluation covered Qwen3.8-Flash-Next, DeepSeek-v4.1-Flash, GLM-5.3, and Kimi-K3 on all 3,420 histories and 171 workload bins in StateTracking. Our main experiment also evaluated GPT-6-Astra on stratified samples.

3.1 Experimental Setup

Models and inference. We evaluated Qwen3.8-Flash-Next (Qiu et al., 2026), Kimi-K3 (Kimi Team et al., 2026), GLM-5.3 (Zeng et al., 2026), DeepSeek-v4.1-Flash (Xu et al., 2026), and GPT-6-Astra (OpenAI, 2026). The specified inference protocol used official model APIs with the highest supported reasoning effort and default values for all other parameters. For GPT-6-Astra, we sampled histories uniformly without replacement within each workload bin: two per bin for Wikipedia Article and Talk (54 each), three for LOBSTER and DNS (135 each), and five for Jira (135).

Tasks and prompting. The common task interface supplied the initial state, ordered history, and output instructions without a final key list, external retrieval, persistent memory, or executable tools. We used identical prompts across models. Wikipedia used natural-language editing descriptions, Jira used issue-history descriptions, and LOBSTER/DNS used the released JSON inputs. The latter retained auxiliary per-key queries, but the reported scores evaluated only their complete-state component. One final answer was scored per task; reasoning text was excluded. Inputs were kept complete. All tasks remained in the denominator, including missing answers, generation failures, and inputs that exceeded the context limit.

Scoring. For normally completed answers requiring JSON, we extracted the first JSON block and applied the dataset scorer. We neither repaired malformed JSON nor selected a later block based on its contents. Across all five datasets, we measured complete-state exact match (EM) as the percentage of tasks whose valid answers contained exactly the reference keys and their complete final values, irrespective of entry order. We ignored case when comparing Jira priority labels. Missing, duplicate-key, or incomplete outputs counted as incorrect.

We also evaluated key recovery independently of values. Key-set EM measured exact agreement with the reference key set.

3.2 Main Results

Complete-state EM was our primary metric because it required both the final keys and their complete values to be correct. Kimi-K3 achieved the highest complete-state EM among the four open-weight models on all five datasets in Table 4, while GPT-6-Astra scored higher on its stratified samples. The largest gap was on Wikipedia Talk: GPT-6-Astra achieved 94.44% on its stratified sample, versus 21.48% for Kimi-K3. Kimi-K3's 88.52% key-set EM further showed that correct keys did not ensure complete value recovery.

Table 4: Complete-state EM / Key-set EM (smaller type) on StateTracking (%).

Model	Wikipedia Article	Wikipedia Talk	Jira	LOBSTER	DNS
Qwen3.8-Flash-Next	57.22 / 93.15	10.56 / 83.70	52.96 / 69.81	58.44 / 61.89	47.22 / 49.22
Kimi-K3	88.33 / 97.78	21.48 / 88.52	85.93 / 88.15	69.44 / 72.00	52.89 / 54.44
GLM-5.3	48.89 / 90.93	11.67 / 77.78	55.56 / 62.04	38.44 / 41.44	35.78 / 38.00
DeepSeek-v4.1-Flash	80.37 / 96.85	18.15 / 95.93	66.48 / 75.74	61.44 / 63.11	49.33 / 50.22
GPT-6-Astra	92.59 / 98.15	94.44 / 100.00	94.07 / 94.81	97.78 / 97.78	89.63 / 90.37

3.3 Effects of State Size, History Length, and Deletion Frequency

For Qwen, DeepSeek, GLM, and Kimi-K3, we compared bins that shared two coordinate settings and differed in the third. Figure 2 reported the change in EM from the lowest to highest setting, averaging equally over the other two coordinates. All bins contained 20 histories. Wikipedia comparisons matched coordinate intervals, whereas the record grids used prescribed values or tolerance bands. Appendix A.4 provided the complete joint-bin results.

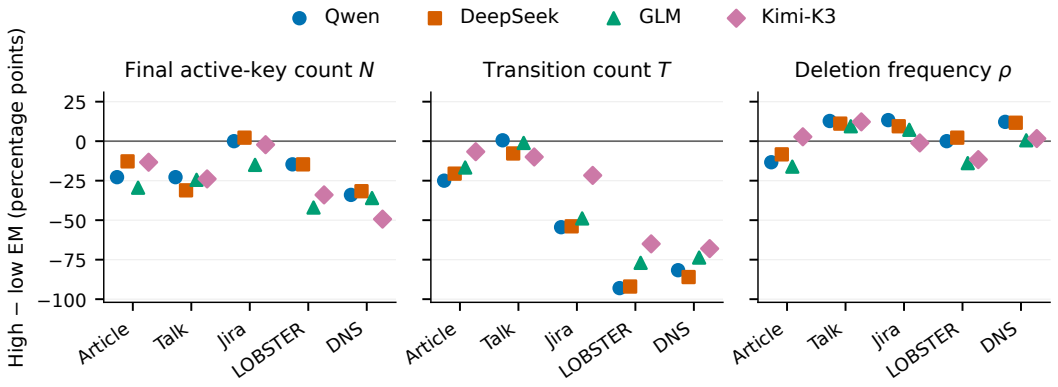


Figure 2: Changes in complete-state EM from the lowest to highest setting of each workload factor, with equal weights over the other two coordinates. Negative values indicated lower EM at the higher setting. Levels were specific to each dataset; all outcomes remained in the denominator.

Larger final states increased the requirement for joint correctness. On Wikipedia Talk, DeepSeek complete-state EM fell from 35.00% to 3.89% between the lowest and highest N settings, while Kimi-K3 fell from 35.56% to 11.67%. Recovering a larger final state required more key–value pairs to be correct simultaneously.

Longer histories exposed both content and output failures. At fixed N and ρ settings, both Qwen and Kimi-K3 complete-state EM declined from the lowest to highest T in every Jira, LOBSTER, and DNS bin pair. On LOBSTER, Qwen EM fell by 93.00 percentage points, accompanied by a 61.33-point increase in invalid or missing answers; DeepSeek and GLM showed the same pattern. Qwen's complete-state EM remained nearly unchanged on Wikipedia Talk. Kimi-K3 lost 10.00 percentage points. The effect of history length therefore varied across models and datasets and included failures to produce valid, complete answers.

Higher deletion frequency did not necessarily make reconstruction harder. All four models gained EM on Wikipedia Talk between the lowest and highest ρ settings. On Wikipedia Article, Qwen, DeepSeek, and GLM lost EM, whereas Kimi-K3 gained 2.78 percentage points. Across these bins, the median total reference-text length fell from 36.5k to 14.7k characters in Wikipedia Talk but rose from 11.9k to 18.3k in Wikipedia Article. These directions held in eight of nine Wikipedia Talk and all nine Wikipedia Article comparisons at the same N and T settings. Shorter final text was therefore a possible contributor to the gains on Wikipedia Talk despite the more frequent deletions. StateTracking's joint bins and complete reference states made this variation in reconstruction demand observable alongside deletion frequency. LOBSTER and DNS provided complementary comparisons with final N fixed: raising ρ at fixed T replaced overwrites with deletion–insertion pairs. Text length, operation mix, and source composition remained relevant to interpreting these trends.

3.4 Error Analysis

Figure 3 summarized the outcome categories. We related answer errors to the supplied history by tracing removed keys, superseded values, and content that remained valid through later edits. These witnesses distinguished failures of updating, preservation, and invalidation, but could overlap within one answer. Text-fidelity and output-format errors were assessed separately.

Correct keys did not establish complete final text. On Wikipedia Talk, 395 Qwen, 420 DeepSeek, 357 GLM, and 362 Kimi-K3 answers recovered every final key but contained incorrect values, out of 540 histories per model. These gaps made membership recovery and value reconstruction separately observable within the same answer.

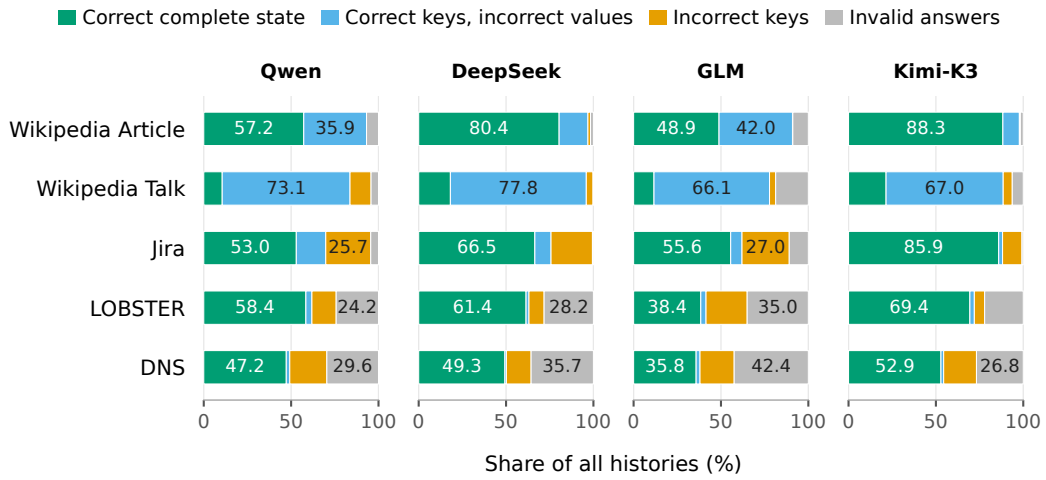


Figure 3: Outcomes on StateTracking for Qwen3.8-Flash-Next, DeepSeek-v4.1-Flash, GLM-5.3, and Kimi-K3. Each bar included all histories: 540 for Wikipedia Article, Wikipedia Talk, and Jira, and 900 for LOBSTER and DNS. Incorrect-key and incorrect-value categories required valid answers. Invalid answers included malformed outputs, absent or incomplete generations, and context-limit failures.

Case study: linking errors to specific updates. In one Wikipedia Talk history ($N = 8, T = 139$, $\rho = 0.647$), Qwen returned the complete earlier text of *Extra graphs* from Talk:2017 in spaceflight, omitting a later comment. It also returned the *Scratchpad* key from Talk:Large language model after that section was deleted. DeepSeek, GLM, and Kimi-K3 handled both keys correctly. These two errors exposed a missed update and a failed exclusion within the same answer.

Case study: earlier values survived later overwrites. In five of Kimi-K3's 23 LOBSTER answers with correct keys but incorrect values, at least one predicted order matched an earlier complete value. For example, an order's remaining size changed from 118 to 18, but Kimi-K3 returned 118. Correct key recovery therefore coexisted with a missed value update.

Case study: preserving valid content through later edits. In a Wikipedia Article history from the *Interrupt* Wikipedia page, GLM omitted a paragraph that remained unchanged throughout the history; Qwen, DeepSeek, and Kimi-K3 returned that section exactly. Requiring every final value made this loss observable alongside missed updates and failed exclusions, without specifying a target key in advance. These traced content errors were distinct from whitespace differences, which accounted for most DeepSeek answers on Wikipedia Talk with correct keys but inexact text, and 299 of the 362 such Kimi-K3 answers.

4 Related Work

4.1 Evolving Memories and Symbolic States

ProPara (Dalvi et al., 2018) tracks entity existence and location in procedural text. LongMemEval (Wu et al., 2025) covers knowledge updates and temporal reasoning. Memora (Uddin et al., 2026) tests direct models and memory agents on simulated long-term interactions, including remaining-collection queries. MemOps (Hao et al., 2026) evaluates memory lifecycle operations and state trajectories, while MEME (Jung et al., 2026) tests tracking, deletion, and dependencies

among facts. Entity Tracking (Kim & Schuster, 2023) and PlanBench (Valmeekam et al., 2023) also include complete symbolic-state prediction.

StateTracking brings these state-maintenance requirements to changing collections of long-form text and structured values. Here, a single answer must recover every final key and its complete value, jointly testing whether updates are incorporated, invalidated content is excluded, and unchanged content remains intact.

4.2 Reconstruction from Observed Revisions

LoCoDiff (Mentat AI Team, 2025) reconstructs a designated surviving file from real Git revisions and balances input lengths across repositories. Our Wikipedia tasks require both identifying the surviving sections and reconstructing their complete text. Evaluating both in the same answer distinguishes membership errors from incorrect values. The supplied histories further allow us to trace returned deleted keys and superseded passages to earlier states, separating these failures from text-fidelity and output-format errors.

The original τ-bench (Yao et al., 2025) checks a final database against a reference, while AppWorld (Trivedi et al., 2024) checks required and allowed database changes. These evaluate results reached through interaction. StateTracking instead supplies a fixed history, leaving action selection and tool execution outside the measured capability.

4.3 Workload Design for State Tracking

Controlled tasks already separate aspects of difficulty. RULER (Hsieh et al., 2024) varies sequence length and task complexity across retrieval, multi-hop tracing, and aggregation tasks. LongProc (Ye et al., 2025) tests integration of dispersed information and long structured generation. Michelangelo (Vodrahalli et al., 2024) varies relevant list operations and background context. Memora (Uddin et al., 2026) also configures operation weights, but its released weekly, monthly, and quarterly histories increase retained entries and transitions together, limiting fixed-size comparisons across horizons.

StateTracking organizes Wikipedia Article, Wikipedia Talk, and Jira histories into predefined joint (N, T, ρ) bins and complements them with prescribed LOBSTER and DNS grids. These support conditional comparisons between similar final-state sizes reached through different update and deletion histories. The common complete-state task connects such workload comparisons across text and structured values, with errors assessed separately for final keys, their values, and output validity.

5 Conclusion

StateTracking evaluates whether accumulated local changes can be turned into a complete final state. Its 3,420 histories combine real document and discussion edits with complementary issue, order, and network-record updates. Each dataset provides 20 histories in every predefined workload bin, allowing comparisons across final active-key count, transition count, and deletion frequency within the same complete current-state reconstruction task. Evaluations on StateTracking show why both membership and complete values matter: exact keys can coexist with incorrect text, while incorrectly returned removed keys and invalid outputs reveal additional failures.

Qwen3.8-Flash-Next, DeepSeek-v4.1-Flash, GLM-5.3, and Kimi-K3 have now been evaluated on all five balanced grids. Their coverage supports conditional diagnosis, while continuous workload variation, shared sources, and output demands limit causal attribution. Extending this evaluation to interactive workflows can connect incomplete final states to downstream decisions.

AI use statement

We used generative AI tools to assist with language polishing of the paper and with writing code for data cleaning during dataset construction. The authors take responsibility for the final content of this work.

REFERENCES

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1595–1604, 2018.

Xixuan Hao, Zeyu Zhang, Zehao Lin, Yihang Sun, Ziliang Guo, Xichong Zhang, Yuxuan Liang, Feiyu Xiong, and Zhiyu Li. MemOps: Benchmarking Lifecycle Memory Operations in Long-Horizon Conversations. *arXiv preprint arXiv:2607.12893*, 2026. URL https://arxiv.org/abs/2607.12893v1.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? *arXiv preprint arXiv:2404.06654*, 2024. URL https://arxiv.org/abs/2404.06654v3.

Bowen Jiang, Zhuoqun Hao, Young Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo Jose Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=6ox8XZGOqP.

Seokwon Jung, Alexander Rubinstein, Arnas Uselis, Sangdoo Yun, and Seong Joon Oh. MEME: Multi-entity & Evolving Memory Evaluation. *arXiv preprint arXiv:2605.12477*, 2026. URL https://arxiv.org/abs/2605.12477v1.

Najoung Kim and Sebastian Schuster. Entity Tracking in Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3835–3855, 2023. doi: 10.18653/v1/2023.acl-long.213. URL https://aclanthology.org/2023.acl-long.213/.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi K3: Open Frontier Intelligence. *arXiv preprint arXiv:2607.24653*, 2026. URL https://arxiv.org/abs/2607.24653.

Mentat AI Team. LoCoDiff: Natural Long Context Code Bench, 2025. URL https://abanteai.github.io/LoCoDiff-bench/. Official benchmark project, May 8, 2025. Accessed September 19, 2026.

Paul Mockapetris. Domain names-implementation and specification, RFC1035. *Information Sciences Institute*, 1987. URL https://www.rfc-editor.org/rfc/rfc1035.html.

OpenAI. GPT-6 Astra: A New Generation of Intelligence. Official blog, September 2026. URL https://openai.com/index/gpt-6-astra/.

Zihan Qiu, Zekun Wang, Xiao Li, Yanpeng Li, Yang Xu, Yixuan Wang, Huaqing Zhang, Rui Men, Bochao Mao, Chengruidong Zhang, et al. On the Design of Qwen3.8-Next Architecture: Evaluation, Efficiency, and Training Stability. *arXiv preprint arXiv:2608.30320*, 2026. URL https://arxiv.org/abs/2608.30320.

S Thomson, C Huitema, V Ksinant, and M Souissi. RFC3596: DNS extensions to support IP version 6, 2003. URL https://www.rfc-editor.org/rfc/rfc3596.html.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16022–16076, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.850. URL https://aclanthology.org/2024.acl-long.850/.

Md Nayem Uddin, Kumar Shubham, Eduardo Blanco, Chitta Baral, and Gengyu Wang. From Recall to Forgetting: Benchmarking Long-Term Memory for Personalized Agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 26814–26841, 2026. doi: 10.18653/v1/2026.findings-acl.1337. URL https://aclanthology.org/2026.findings-acl.1337/.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. PlanBench: An extensible benchmark for evaluating large language models on planning and reasoning about change. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 38975–38987. Curran Associates, Inc., 2023. doi: 10.52202/075280-1693. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7a92bcdede88c7afd108072faf5485c8-Paper-Datasets_and_Benchmarks.pdf.

Kiran Vodrahalli, Santiago Ontanon, Nilesh Tripuraneni, Kelvin Xu, Sanil Jain, Rakesh Shivanna, Jeffrey Hui, Nishanth Dikkala, Mehran Kazemi, Bahare Fatemi, Rohan Anil, Ethan Dyer, Siamak Shakeri, Roopali Vij, Harsh Mehta, Vinay Ramasesh, Quoc Le, Ed Chi, Yifeng Lu, Orhan Firat, Angeliki Lazaridou, Jean-Baptiste Lespiau, Nithya Attaluri, and Kate Olszewska. Michelangelo: Long Context Evaluations Beyond Haystacks via Latent Structure Queries. *arXiv preprint arXiv:2409.12640*, 2024. URL https://arxiv.org/abs/2409.12640v2.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory. In *International Conference on Learning Representations*, pp. 86809–86836, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d813d324dbf0598bbdc9c8e79740ed01-Abstract-Conference.html.

Anyi Xu, B Li, Bangcai Lin, Bing Xue, BingCheng Xian, Bingzheng Xu, Bochao Wu, Bowei Zhang, Boyi Deng, CC Yu, et al. DeepSeek-V4.1-Flash: Pushing the Limits of KV Cache Compression. *arXiv preprint arXiv:2609.19969*, 2026. URL https://arxiv.org/abs/2609.19969.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 9965–10017, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/1b126cc38b8638e07bef37e7b2bb72bf-Paper-Conference.pdf.

Xi Ye, Fangcong Yin, Yinghui He, Joie Zhang, Howard Yen, Tianyu Gao, Greg Durrett, and Danqi Chen. LongProc: Benchmarking Long-Context Language Models on Long Procedural Generation. *arXiv preprint arXiv:2501.05414*, 2025. URL https://arxiv.org/abs/2501.05414v3.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. GLM-5: from Vibe Coding to Agentic Engineering. *arXiv preprint arXiv:2602.15763*, 2026. URL https://arxiv.org/abs/2602.15763.

A Additional Dataset and Evaluation Details

A.1 Wikipedia Construction

Selecting revisions and matching sections. We retrieved Wikipedia article and discussion-page revisions together with their identifiers and timestamps. Temporal selection favored recent edits to reduce potential training-data leakage: Wikipedia Article initialized each page from its last revision at or before December 31, 2025, 23:59:59 UTC, and retained subsequent updates through September 9, 2026, 23:59:59 UTC, while Wikipedia Talk used the latest 50 or 100 revisions available per source page at collection time. We represented the article lead and top-level sections by keys, with their text as values. Talk used discussion sections without the lead. We aligned sections across adjacent revisions using headings and content, excluding ambiguous identity matches. Article kept Wikipedia source text, while Talk used a plain-text representation of the discussion.

Assembling histories and checking references. An Article task followed a continuous revision chain from one page. Its initial state was the first retained page state, and each subsequent revision contributed a group of section changes applied together. A Talk task combined selected portions of real section histories. We recovered each section's starting state from its preceding history, then interleaved the selected edits while preserving their order within that section, as in order-preserving memory-history construction (Jiang et al., 2025). Templates produced paired natural-language and formal inputs with the same history and answer. We extracted complete final states from the corresponding source histories and verified that replaying the public edits reproduced them for all 1,080 tasks. We also checked that the two input representations described the same changes. Reference extraction and public-history replay followed separate paths but shared the same upstream text preparation and section matching.

A.2 Wikipedia Joint Workload Coverage

We selected the dataset-specific intervals in Table 2 using metadata from 2,000 candidate histories per dataset, without model scores. We chose boundaries to support a quota of 20 histories in each joint bin while reducing the need for additional compositions. The boundaries therefore reflect source support rather than equal-width intervals or presumed difficulty thresholds. Within each bin, we prioritized existing evaluation members and used a fixed ordering to select histories.

Wikipedia Article and Talk each contained 540 histories, with 20 histories in each of the 27 joint workload bins.

Table 5: Wikipedia workload statistics, reported as median / maximum. Text lengths are measured in Unicode characters.

Quantity	Article	Talk
Final active keys N	7 / 103	8 / 42
Section transitions T	12 / 1,040	96 / 550
Natural-language prompt characters	60,574 / 5,176,095	272,030 / 1,405,253
Formal prompt characters	47,185 / 3,343,687	225,154.5 / 1,163,312
Answer JSON characters	15,560.5 / 391,083	26,496.5 / 192,136

Table 5 summarizes the final datasets. Talk histories can share source pages and section traces. We retained source-page, revision, and section-trace identifiers for grouped analyses.

In 511 Article and 504 Talk histories, at least one nonempty final body was not contained in any single checked initial or creation body, contiguous literal block, or before/after text of one edit hunk.

A.3 Record Construction and Control Boundaries

A.3.1 LOBSTER

We constructed LOBSTER to vary key turnover while holding the initial and final collection size fixed. This permits evaluating key removal and recovery of final field values as removed keys accumulate, without increasing the number of orders to return. Its synthetic state mapped order IDs to side, integer-valued price, and remaining quantity. An insertion introduced a fresh ID, a deletion removed an active key, and an overwrite replaced the value stored at that step. In the released trajectories, overwrites strictly reduced positive remaining quantity while preserving side and price. Each deletion was followed by a fresh insertion, so the number of active keys remained in $\{N-1, N\}$.

Turnover at a fixed final active-key count. The dataset contained 900 distinct trajectories, with 20 seeds in each bin of $N \in \{16, 64, 256\}$, $T \in \{256, 1024, 4096\}$, and target deletion frequency $\rho \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. Within each N, T condition, we set the deletion count to $D = \mathrm{round}(\rho T)$, used the same number of insertions, and filled the remaining transitions with overwrites. We retained the realized frequency D/T and used $C = D/N$ to describe cumulative turnover relative to endpoint collection size. For example, $N = 16$, $T = 4096$, and $\rho = 0.5$ produce 2,064 distinct historical IDs but only 16 orders in the final state. This condition contains no overwrites: increasing ρ increases the number of departed identities while reducing the overwrite budget.

It therefore compares operation mixtures, not deletion in isolation. Fixing N fixes the number of output orders, not their exact token length.

Paired representations and reference checks. The dataset retained tuple and JSON renderings, with complete field values on each insertion and overwrite. Independent parsing recovered their initial states and ordered changes, and replay recovered all reference final states.

Interpretation of history length. LOBSTER's fresh identifiers and strictly decreasing quantities admit an order-independent solution: subtract every deleted ID from the union of initial and inserted IDs, then select each surviving ID's field values from the entry with the minimum remaining quantity. Thus T measures the supplied operation workload, not a lower bound on sequential reasoning depth.

A.3.2 DNS

We extended the same workload grid to DNS using records with type-dependent fields. A key identified a synthetic domain name. Its value contained a record type, time-to-live (TTL), and target, plus a priority field for MX records. The five types were A, AAAA, CNAME, MX, and TXT (Mockapetris, 1987; Thomson et al., 2003). Insertions introduced fresh names, deletions removed entire records, and overwrites replaced the complete value while preserving its type. TTL, target, and MX priority could change separately or together, requiring the final answer to respect each record's schema. This implemented explicit record-map maintenance, without DNS resolution or automatic TTL expiration.

The dataset contained 900 trajectories over the same 45 N, T, ρ bins as LOBSTER, with 20 seeds per bin. Paired trajectories preserved their initial and final key sets and complete operation/key-access sequences under a bijective renaming of identities. Values and update rules differ, so a paired dataset contrast does not isolate schema complexity. Each deletion was followed by a fresh insertion, retaining the number of active keys in $\{N-1, N\}$. Some DNS overwrites left all fields unchanged. They remained visible and counted toward T, so T denotes declared operations rather than only effective value changes. All tuple/JSON renderings independently decoded and replayed to the reference final states.

A.3.3 JIRA

We constructed Jira to ground key removal in a workflow view: an issue may continue to exist while its key leaves the state map. The state mapped issue keys to priority, assignee, and status. Creation or reopening inserted an issue key, field changes overwrote its value, and transition to Resolved, Closed, or Done removed its key from the map. Reopening could insert the same key again, unlike LOBSTER's fresh-ID turnover. An issue created in a final status did not add an active key unless it was subsequently reopened. These rules separated membership in the requested view from changes to the issue's contents.

Mapping real change histories to the operator algebra. We derived events from cached public issue histories of Apache projects. For each issue we retained changelog entries for the single-valued tracked fields, priority, assignee, and status; changes to other fields, links, and comments did not enter the tracked-state history. Starting from the snapshot observed at collection time, we recovered the fields at creation by reversing the recorded tracked-field changes. We then applied the rules for active keys above and merged issue-level events in timestamp order. As a consistency check, the window-end reference was compared with replay of the corresponding per-issue histories to the same endpoint.

Configuring windows from real timelines. We selected time windows over cached project blocks to cover different numbers of active keys and history lengths. A window started with the issues open within its cached block at that time; subsequent creations entered at their recorded timestamps. Initial, peak, and final collection sizes could therefore differ. Non-insertion events addressing an absent key, such as a further final-status update on an already closed issue, were dropped before computing T and ρ. Here T counts the retained canonical transitions, including overwrites that left all tracked values unchanged.

We targeted final active-key counts $N \in \{64, 128, 256\}$, transition counts $T \in \{256, 512, 1024\}$, and deletion frequencies $\rho \in \{0.2, 0.3, 0.4\}$. Bins allowed deviations of $\pm 10\%$ in final N and T, $\pm 50\%$ in initial and peak size relative to the target N, and ± 0.03 in ρ. Within each bin, pairwise Jaccard similarity was required to be below 0.5 for the windows' issue footprints, each comprising initial keys and keys touched by events. This limited overlap within a bin without establishing independence across bins.

Synthetic completion of sparse bins. We supplemented sparse bins with synthetic windows that shared the event algebra and schema and retained provenance labels. Each synthetic window fixed its final active-key count to the bin target and sampled initial-to-final and peak-to-final size ratios from real donor windows at the matching deletion level. These ratios were clipped to the admissible bands and checked for schedule feasibility. We set the insertion count to the deletion count plus the change from initial to final active-key count, then used overwrites for the remaining transitions. Values and issue-summary text came from Jira-style pools, and synthetic issue keys used namespaces disjoint from the real corpus.

The resulting dataset contained 455 real and 85 synthetic windows, totaling 540. Synthesis supplemented six underfilled bins, increasing the number meeting the 20-window target from 21/27 to 27/27.

Readable updates and reference checks. We expressed the initial fields and state-changing events as natural language while preserving explicit operation semantics. Some canonical overwrites that left all tracked values unchanged were omitted from the rendered text but still counted toward T; it is therefore not a count of visible update statements. Every status surface carried its canonical status token, including uncommon statuses. Canonical replay recovered all reference final states, and checks against the canonical changes confirmed that every required field value appeared in the public text. This validation used the canonical changes rather than an independent parser of the natural-language history.

A.4 Supplementary Results

Joint workload results. Figures 4 and 5 reported complete-state EM in every workload bin for the four models evaluated on all histories.

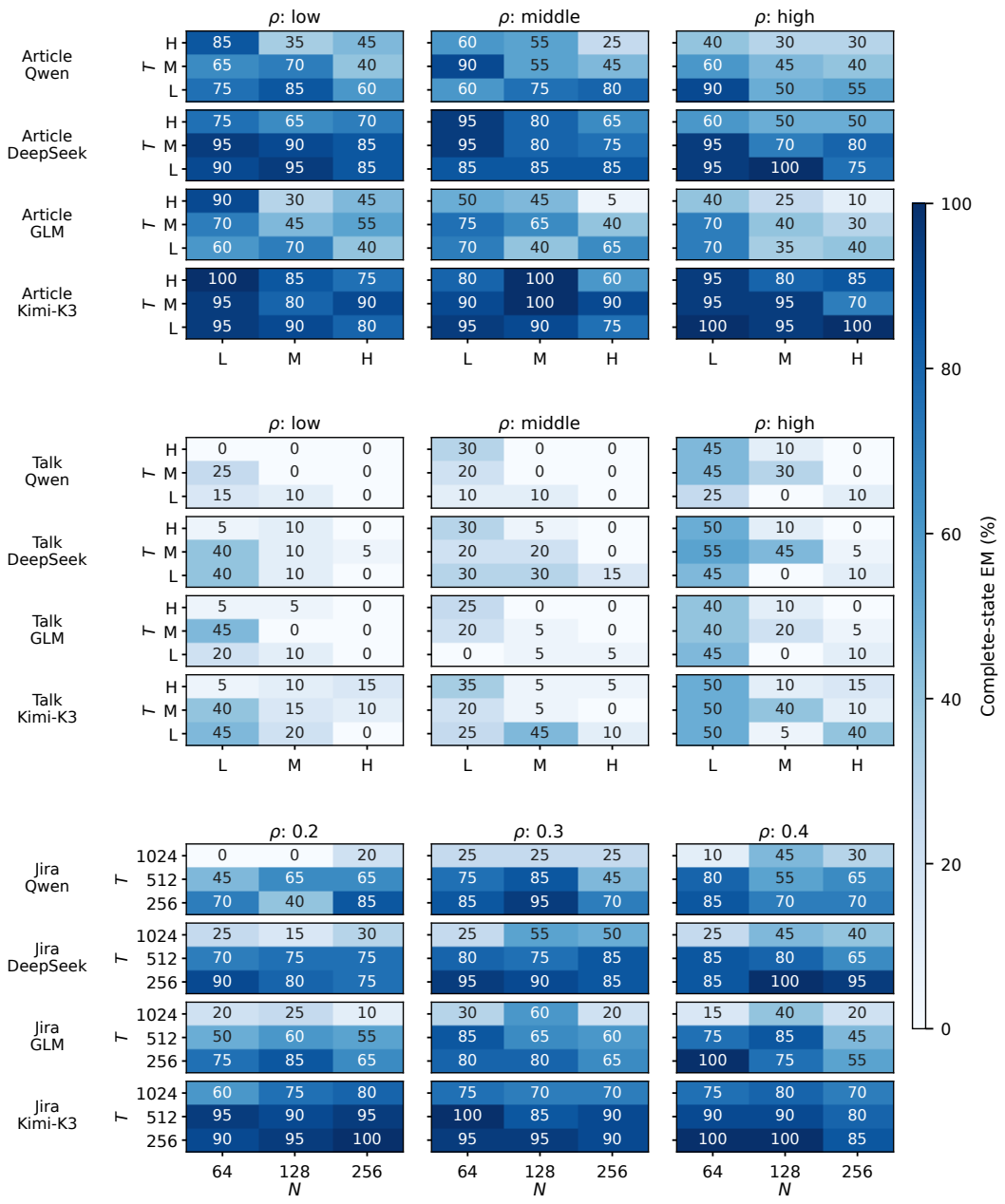


Figure 4: Complete-state EM (%) across all joint workload bins in Wikipedia Article, Wikipedia Talk, and Jira. Columns varied ρ; each matrix varied N horizontally and T vertically. For Wikipedia, L/M/H denoted low, medium, and high; Table 2 specified each factor's numerical ranges separately for Article and Talk. Jira axes showed target values. Each bin contained 20 histories.

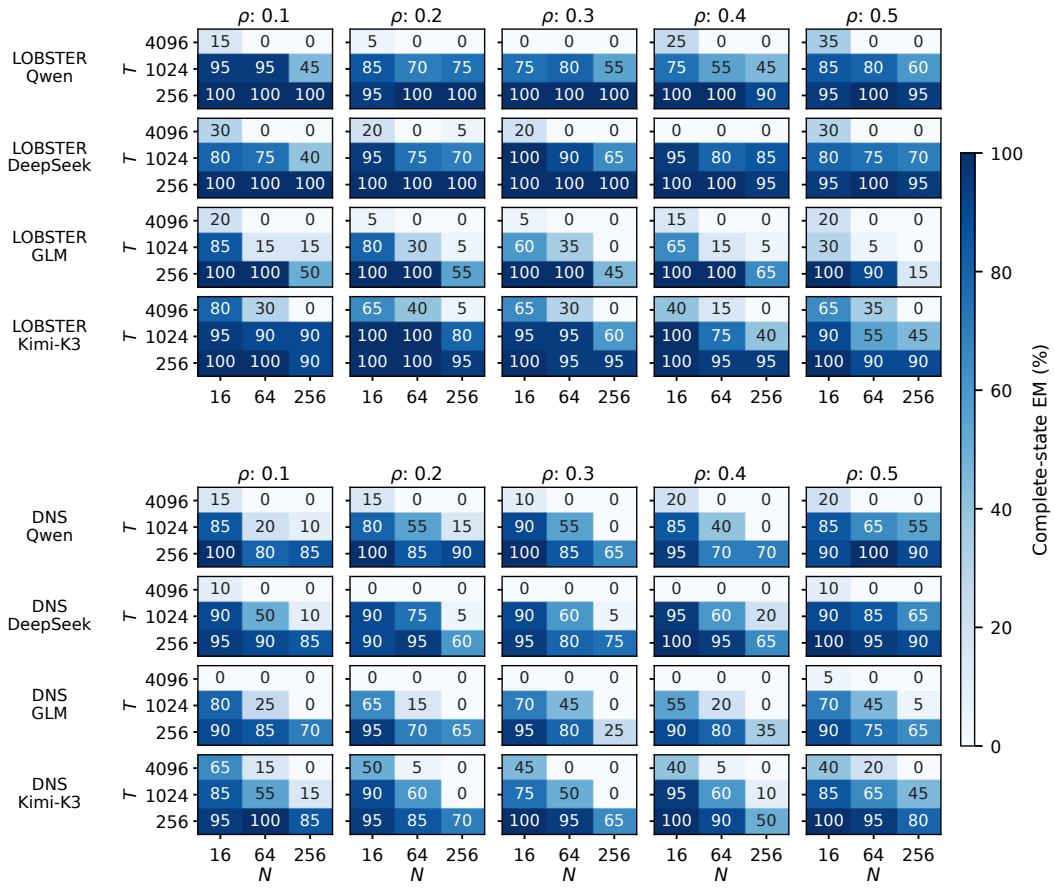


Figure 5: Complete-state EM (%) across all LOBSTER and DNS workload bins. Columns varied the prescribed ρ, and each matrix varied N horizontally and T vertically. Every bin contained 20 histories. Matching settings across datasets did not remove differences in values, input lengths, or update rules.