EPISTEMIC MEMORY FAILURES IN LONG-FORM NARRATIVE AGENTS: A DEPLOYMENT STUDY

Chen Xiwei
962557130s@gmail.com

ABSTRACT

We report findings from deploying an LLM-based narrative agent across 90 chapters of novel generation (180,000+ tokens over 3 months). We identify a previously under-discussed failure mode: **known-information forgetting**—where characters redundantly ask about or rediscover facts they already learned in previous chapters. This failure is distinct from hallucination (facts are correct) and world-state inconsistency (world is consistent); rather, it reflects a mismatch between world state and *character epistemic state*. We trace the root cause to naive recency-based context injection, which systematically excludes mid-chapter key facts. We propose **Key Facts Injection**: extracting semantically important facts from episodic memory and injecting them with explicit "already knows" markers. This simple intervention reduced known-information forgetting by 73% in our deployment. We share our memory architecture design (episodic/semantic/working memory) and lessons learned, hoping to inform future agent memory systems where tracking *what characters know* matters as much as *what happened*.

1 INTRODUCTION

Long-form narrative generation presents unique challenges for LLM-based agents. Unlike dialogue systems or single-document generation, a serialized novel requires:

- **Cross-chapter consistency**: Events in Chapter 90 must not contradict Chapter 1
- **Character epistemic tracking**: What does each character *know* at each point?
- **Causal coherence**: Actions must follow from established motivations
- **Foreshadowing management**: Planted hints must eventually pay off

Standard retrieval-augmented generation (RAG) (Lewis et al., 2020) addresses factual grounding but not these narrative-specific requirements. Generative agents (Park et al., 2023) demonstrate memory streams for behavioral simulation, but focus on short interactions rather than long-form coherent narratives.

We present a memory architecture developed through 90 chapters of production novel generation (approximately 180,000 Chinese characters). Rather than proposing a novel architecture, we report deployment experience and identify design patterns that may generalize:

1. **Failure mode identification**: We characterize "known-information forgetting" as a distinct failure where characters forget previously established facts—different from hallucination or world-state inconsistency
2. **Root cause analysis**: We trace this to recency-based context injection missing mid-chapter key facts
3. **Simple mitigation**: Key Facts Injection with explicit epistemic markers reduces incidents by 73%
4. **Design lessons**: We share our three-tier memory architecture and token budget system for practitioners building similar agents

2 Related Work

Memory in LLM Agents. Recent surveys (Zhang et al., 2024) categorize agent memory into short-term (context window) and long-term (external storage). MemoryBank (Zhong et al., 2024) proposes memory consolidation inspired by human cognition. Voyager (Wang et al., 2023) uses skill libraries as procedural memory. These works focus on task completion; we focus on narrative coherence where epistemic state tracking is critical.

Cognitive Architectures. Sumers et al. (2024) propose cognitive architectures for language agents drawing on psychological models. We adopt their episodic/semantic/working memory distinction but highlight a gap: existing frameworks track *world state* but not *character knowledge state*—a distinction crucial for narrative agents.

Explicit vs. Parametric Memory. A key workshop theme is the interplay between explicit (symbolic/textual) and parametric (in-weights) memory. Our Key Facts Injection demonstrates that when parametric memory fails to preserve epistemic state, explicit memory with instruction-level constraints provides a reliable fallback—without fine-tuning or weight modification.

3 Problem: Known-Information Forgetting

3.1 Definition

We define **known-information forgetting** as: generating content where a character asks about, discovers, or reacts with surprise to information they already learned in a previous chapter.

This differs from hallucination (Ji et al., 2023)—the facts are correct, but the character's epistemic state is wrong. It also differs from simple inconsistency—the world state is consistent, but character behavior violates established knowledge.

3.2 Motivating Example

In our deployed system, Chapter 89 contains:

> *Su Wanqing said: "The operation codename is 'Wheelchair and Bait'—I named it myself."*

In the initially generated Chapter 90:

> *Chen Mo looked at the mission briefing. "Wheelchair and Bait"—who came up with this codename? It sounded like mockery.*

The protagonist asks about information explicitly told to him one chapter earlier. Readers of serialized fiction notice such errors immediately, damaging narrative credibility.

3.3 Root Cause Analysis

Standard context injection uses recency-based selection: inject the last k tokens from previous chapters. With typical token budgets ($k \approx 400$ characters), mid-chapter events are systematically excluded.

In our example, the codename revelation occurs at line 156 of a 280-line chapter. The last 400 characters capture only the chapter's conclusion—a scene transition—missing the key dialogue entirely.

4 Architecture

4.1 Three-Tier Memory

Our system implements three memory tiers inspired by cognitive architectures (McClelland et al., 1995; Sumers et al., 2024):

Table 1: Memory architecture overview

Tier	Contents	Update Frequency
Episodic	Event graph (who, what, when, where)	Per chapter
Semantic	Character states, world facts, relationships	Per chapter
Working	Context window for generation	Per generation

Episodic Memory stores events as a directed graph. Each node contains: event ID, chapter, participants, location, description, effects, and an `is_irreversible` flag for critical plot points. Edges represent causal and temporal relationships.

Semantic Memory maintains per-character state files including: current location, mental state, active goals, relationships (with address terms), and power level. A separate world state tracks global facts, timeline, and environmental conditions.

Working Memory is constructed per-generation by selecting relevant content from episodic and semantic memory, subject to token budget constraints.

4.2 Token Budget and Context Selection

Context injection operates under strict token budgets (6,500 total; see Appendix A for detailed allocation). The Context Selector retrieves relevant content through multiple strategies:

The Context Selector retrieves relevant content through multiple strategies:

1. **Recency**: Last N events (temporal continuity)
2. **Participant overlap**: Events involving current chapter's characters
3. **Location match**: Events at current setting
4. **Irreversibility**: All events marked `is_irreversible`
5. **Semantic similarity**: Vector search for thematically relevant events

The semantic search component uses sentence embeddings to retrieve events relevant to the current chapter outline, enabling recall of distant but thematically connected plot points.

5 Method: Key Facts Injection

5.1 Extraction

For each previous chapter, we extract "key facts" from the event graph:

1. **Named entities**: Codenames, titles, specific terms introduced
2. **Relationship changes**: New alliances, revelations, agreements
3. **Information transfers**: Dialogue where Character A tells Character B something

Extraction targets the `relation_changes.evidence` field in event nodes, which stores verbatim dialogue establishing new information.

5.2 Injection Format

Key facts are injected with explicit epistemic warnings:

```
## PREVIOUS CHAPTER KEY FACTS (Ch.89)
WARNING: Protagonist ALREADY KNOWS these facts.
DO NOT have him ask about or rediscover them:
- Operation codename "Wheelchair and Bait"
  (told by Su Wanqing)
- The frequency matching mechanism
```

The explicit "ALREADY KNOWS" framing leverages instruction-following capabilities to prevent redundant discovery. This is distinct from standard RAG, which retrieves *relevant* information; we retrieve *already-known* information with epistemic framing.

5.3 Tiered Character Injection

Supporting characters receive tiered context injection based on narrative importance (high: full state; medium: location and key traits; low: name only). Importance scores derive from character design completeness and plot relevance.

6 Deployment Results

6.1 System Overview

We deployed the memory architecture in a production novel generation system:

- **Scale**: 90 chapters, $\sim$180,000 Chinese characters
- **Duration**: 3 months of incremental generation
- **Model**: DeepSeek-V3 (primary), Qwen-2.5-72B (backup)
- **Human oversight**: Author review and editing per chapter

6.2 Known-Information Forgetting

Table 2: Known-information forgetting incidents by condition

Condition	Chapters	Incidents	Rate
Baseline (recency only)	1–45	12	26.7%
+ Key Facts Injection	46–90	3	6.7%

Key Facts Injection reduced known-information forgetting by 73% (Fisher's exact test, $p < 0.05$). The remaining 3 incidents occurred when: (1) facts were established 3+ chapters prior, outside the extraction window; (2) information was implicit rather than explicitly stated; (3) multiple characters were present, creating ambiguity about who "knows" what.

6.3 Other Metrics

Beyond known-information forgetting, we tracked other consistency metrics (see Appendix B): timeline contradictions (4), character location errors (7), relationship inconsistencies (5), and dropped foreshadowing (2). The foreshadowing alert system detected cases where planted plot threads exceeded their expected resolution window.

Semantic search ablation (Appendix C) showed 62% more relevant events retrieved with vector search, with average source distance increasing from 3.1 to 12.4 chapters.

7 Discussion

Epistemic State vs. World State. Our central finding is that narrative agents must distinguish *what happened* from *what characters know happened*. This distinction may generalize beyond narrative to any agent where multiple entities have partial observability.

Explicit Memory as Fallback. Key Facts Injection demonstrates that explicit memory with instruction-level constraints can compensate for parametric memory failures—without fine-tuning. This suggests a design pattern: use explicit memory not just for retrieval, but for *epistemic constraints*.

Limitations. (1) Single project evaluation limits generalizability. (2) Per-character epistemic tracking remains manual. (3) The 73% improvement leaves room for further gains.

8 CONCLUSION

We reported deployment experience from a 90-chapter narrative agent, identifying known-information forgetting as a distinct memory failure mode. The root cause—recency-based context injection missing mid-chapter facts—suggests that standard RAG approaches are insufficient for agents requiring epistemic state tracking. Key Facts Injection, a simple explicit memory intervention, reduced incidents by 73%. Our findings suggest that agent memory systems benefit from distinguishing *what happened* from *what each entity knows happened*—a distinction that may generalize beyond narrative to multi-agent and partial-observability settings.

REFERENCES

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474, 2020.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Joon Sung Park, Joseph C O'Brien, Carrie J Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442*, 2023.

Theodore R Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*, 2024.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Zeyu Zhang, Xiaohe Zhang, Tong Zhang, Dawei Zhu, Jingbo Chen, Jianye Shao, and Hengshu Zhu. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. *arXiv preprint arXiv:2305.10250*, 2024.

A Token Budget Allocation

Table 3: Token budget allocation (total: 6,500 tokens)

Category	Budget	Purpose
MC State	800	Protagonist's current state
Supporting Characters	600	Tiered injection by importance
Recent Events	1,200	Last N events from graph
Semantic Events	500	Vector-retrieved relevant events
Foreshadowing	500	Pending plot threads
World State	400	Environmental context
Character Designs	600	Personality, speech patterns
Milestones	500	Upcoming plot requirements
Reserve	400	Formatting overhead

B Consistency Metrics

Table 4: Consistency metrics across 90 chapters

Metric	Count	Notes
Timeline contradictions	4	Fixed by timeline_plan injection
Character location errors	7	Reduced after state sync
Relationship inconsistencies	5	Address term mismatches
Dropped foreshadowing	2	Detected by alert system

C Semantic Search Ablation

Table 5: Semantic search ablation (chapters 70–90)

Retrieval Method	Relevant Events Retrieved	Avg. Distance
Recency + Participant only	4.2	3.1 chapters
+ Semantic search	6.8	12.4 chapters