**AI Dungeon Master: A Memory-Driven Multi-Agent Storytelling System**

**Abstract**

This paper presents AI Dungeon Master, a generative role-playing engine for interactive storytelling. It combines a multi-agent language model, adaptive creativity control, and hierarchical memory to maintain consistent worldbuilding and character evolution. Using Groq API for reasoning, ChromaDB for vector memory, and Sentence Transformer embeddings, it delivers contextually consistent and emotionally adaptive narratives. Implemented in Python, it features a console interface and developer debug tools for evaluation.

**1. Introduction**

Large Language Models (LLMs) excel at creative text but often lack long-term coherence. AI Dungeon Master addresses this by combining three specialized agents—Dungeon Master, Memory Manager, and Lore Talker—with dynamic temperature control and hierarchical vector memory. This architecture enables adaptive, explainable, and scalable storytelling across multi-turn interactions.

**2. System Design and Architecture**

**2.1 Multi-Agent Design**

The core framework is structured around three specialized agents that interact in sequence each turn:

| **Agent** | **Primary Role** | **Core Functionality** |
| --- | --- | --- |
| Dungeon Master (DM) | Narrative generator | Produces creative storylines and manages real-time player interaction |
| Memory Manager | Cognitive memory controller | Encodes, retrieves, and classifies events into world, NPC, and location memories using vector embeddings |
| Lore Talker | Consistency verifier | Ensures generated responses align with prior lore and factual continuity |

**Workflow per turn:**

1. Player input is processed by the Memory Manager, which queries ChromaDB for relevant stored events
2. The Lore Talker filters and validates retrieved context for consistency
3. The Dungeon Master integrates this context with player input to generate the next narrative segment through Groq's llama-3.1-8b-instant model
4. Key new facts are extracted and re-encoded by the Memory Manager for future recall

This design creates a cooperative reasoning loop, where creativity (DM), factual memory (Memory Manager), and world consistency (Lore Talker) function as complementary modules.

**2.2 Dynamic Temperature Control**

Temperature governs the degree of randomness in LLM sampling. A static value often results in monotonous or incoherent tone shifts. To counter this, AI Dungeon Master employs context-sensitive dynamic temperature control.

The system automatically adjusts the model's temperature based on scene type and emotional intensity detected from player input:

| **Context Type** | **Detected Keywords** | **Temperature** | **Response Behaviour** |
| --- | --- | --- | --- |
| Lore / Recall | "remember", "who was", "when" | 0.3 | Logical and factual |
| Normal Dialogue | Default case | 0.7 | Balanced storytelling |
| Action / Fear Scene | "attack", "run", "danger" | 1.0 | Expressive, dramatic |

This enables emotionally adaptive storytelling — calm in analytical contexts and highly creative during intense narrative scenes.

**3. Memory System Design**

**3.1 Hierarchical Memory Classification**

The project introduces a three-tiered RAG (Retrieval-Augmented Generation) architecture, where memories are organized by relevance scope:

| **Memory Type** | **Description** | **Collection Example** |
| --- | --- | --- |
| World Memory | Global lore, quests, geography | World memory |
| NPC Memory | Per-character interactions, trust, or relationships | NPC goblin king, NPC wizard |
| Location Memory | Environmental or event-specific data | forest, temple, castle |

Each entry is embedded using sentence-transformers and stored in ChromaDB with metadata:

Example-

json

{

"text": "Player gave the silver key to Goblin King.",

"npc": "Goblin King",

"location": "Forest Ruins",

"importance": 0.9,

"timestamp": 1738865200.0

}

Retrieval is performed using a contextual scoring function:

with tuneable weights (α=0.6, β=0.3, γ=0.1).

This ensures semantically relevant, recent, and significant facts dominate narrative recall.

**3.2 NPC Personality and Evolution**

Each NPC maintains a personality vector encoding emotional traits such as friendliness, greed, wisdom, and fear. For instance:

{"friendly": 0.7, "greedy": 0.2, "fearful": 0.4, "wise": 0.6}

After every interaction, these attributes are updated through sentiment analysis of both player input and the DM's response. Over time, NPCs exhibit adaptive behaviour

This mechanism provides long-term relationship continuity, significantly improving narrative realism.

**4. Smart Memory Debug Console**

A core innovation is the Smart Debug Console, which provides live introspection into the AI's reasoning and memory operations.

At runtime, it displays:

* Retrieved memories and similarity scores
* Newly added or summarized facts
* Active NPC memory states
* Short-term vs. long-term memory counts

Example Output:

Retrieved Memories:

"Goblin King trusts player" [score: 0.91]

"Player recovered silver key" [score: 0.84]

Added Memory: "Player found the Heart of Ember fall."

Stats: 52 memories | 4 NPC collections | Short-term turns: 5/5

This offers both transparency and evaluation traceability, allowing direct verification of RAG pipeline operations.

**6. Evaluation and Testing Methodology**

Our evaluation framework focuses on three core areas:

The system is evaluated on three key aspects: **memory effectiveness** (tracking contextual retrieval via the debug console), **multi-agent coordination** (monitoring agents’ role integration for coherent narratives), and **adaptive behaviour** (testing creativity control and NPC personality evolution).

This methodology emphasizes practical demonstration of capabilities through transparent observation, focusing on creating an AI Dungeon Master that maintains engaging, consistent, and memorable interactive storytelling experiences.

**7. Conclusion**

AI Dungeon Master demonstrates an effective fusion of retrieval-augmented generation, multi-agent reasoning, and personality-driven dialogue modelling.

By introducing hierarchical memory, adaptive creativity control, and transparent debugging, the system maintains narrative coherence over long interactions while exhibiting evolving character dynamics.