3.1 Retrieval from the Conversation

Indexing the Conversation: After each user–assistant turn (Figure 2, top), we apply Qwen2.5-32B-AWQ (Team, 2024) with the prompt in Listing 40 (Appendix G) to extract key–value pairs and a summary of the interaction. Keys represent entities and values capture attributes or descriptive details, providing fine-grained, event-level indices analogous to hippocampal memory traces (Teyler & DiScenna, 1986). These key–value pairs and summaries are embedded using the BAAI/bge-small-en-v1.5 embedding model (of Artificial Intelligence, 2023) and stored in a vector database as keys, while the original dialogue segments are kept as values to ensure faithful grounding.

Retrieval from the Index: To retrieve information from the conversation as episodic memory, we embed the question x using the same embedding model and compare it against the stored keys in the index, and the original dialogue segments corresponding to the top k nearest neighbors are returned.

3.2 Scratchpad Formation and Utilization

Construction: In addition to episodic memory (Figure 2, middle pathway), we build a higher-level representation that preserves information beyond individual dialogue events. It integrates semantic knowledge (facts and concepts), autobiographical details (life events), prospective memory (future intentions), and contextual metadata (time, place, acquisition context) (Binder & Desai, 2011). For each dialogue pair, we use Qwen2.5-32B-AWQ with the prompt in Listing 41 (Appendix G) to reason over the current and preceding turn and extract salient content. The resulting "scratchpad" is iteratively merged with earlier versions; once content exceeds a 30K-token threshold—substantially shorter than the raw conversation—it is compressed into a 15K-token summary by GPT-4.1-nano using the prompt in Listing 42. This process maintains efficiency and long-term coherence, analogous to the gradual abstraction of semantic memory in humans. Unlike the episodic index, the scratchpad is not stored in a retrieval database but is provided directly as contextual input during inference.

Filtering Scratchpad (function f): During inference, the scratchpad is selectively filtered with respect to the question. It is first divided into semantically coherent chunks using *semantic chunking*. [2] Each chunk is evaluated by Qwen2.5-32B-AWQ with the prompt in Listing 43 (Appendix G), which assigns a binary relevance label (`yes`/`no`). Only the chunks judged relevant are retained, producing a condensed representation of scratchpad that is passed to the response generator.

4 Experiments

4.1 Experimental Setup

Baselines: We evaluate our approach against two types of baselines: long-context LLMs and a RAG method. For long-context LLMs, the entire conversation history is provided, followed by the probing question. We include two proprietary LLMs (*GPT-4.1-nano*, *Gemini-2.0-flash*, both 1M context). and two open-source models (*Qwen2.5-32B-AWQ*, *Llama-4-Maverick-fp8*). For long-context experiments, *Qwen2.5-32B-AWQ* is evaluated with a 128K context length, while for the RAG baseline and our proposed method a 32K context length is used. At the 10M-token, since none of the four models support this length, they are evaluated on the largest recent dialogue segment fitting their window.[3] For RAG baselines, each user–assistant turn pair is treated as a document, embedded and stored in a vector database. At inference, the top five most similar documents are retrieved and passed to the LLM using the prompt in Listing 44 (Appendix G).

Inference Setup: For inference, we use Nucleus (Holtzman et al., 2020) with temperature 0, except for conversation plan, user-turn, and assistant-turn generation, where temperature is 0.1 to encourage diversity. All open-source LLMs are served via VLLM for efficient inference. For Llama3.3-70B, we set the maximum output length to 6K tokens during user-turn generation, while

[2]`SemanticChunker` in LangChain is used, which segments text into variable-length passages based on semantic rather than fixed token windows.

[3]Among available models, only *Llama-4-Scout* supports 10M-token context windows; however, due to its extreme computational requirements, we were unable to include it in our experiments.