Hypomnemata and the New Industrialization of Memory in GenAI

James E. Dobson[1]

Abstract: The operation of contemporary commercial GenAI chatbots was transformed in early 2025 through the quietly introduced "memory" feature. Memory, in this context, refers to the inclusion of selected summaries of prior interactions alongside the current prompt in the model's context window, producing the appearance of persistent state in a fundamentally stateless architecture. Through the industrial management techniques directed toward the textual objects that I refer to collectively as memory-as-context, chatbot applications implementing this memory feature blur the previously clear distinction between in-context learning and in-weights learning. By continually retrieving and reinserting prior interactions into new contexts, these applications create the appearance of ongoing adaptation and persistent personal memory, even though the underlying language model remains unchanged. This manufactured continuity encourages the perception of a chatbot personality, intensifies anthropomorphic interpretations of model behavior, and enables the construction of increasingly detailed user profiles that can become objects of information extraction and monetization. This essay takes up two case studies, the research prototype MemoryBank and the open-source application OpenClaw, to trace the common techniques and strategies through which GenAI-enabled memory-as-

[1] **James E. Dobson** is Associate Professor of English and Creative Writing at Dartmouth College, where he has been teaching in the humanities since 2012 and researching computational methods since 2003. His interests and publications span a wide range of topics, from accessing computational and data resources on large, distributed computing networks to autobiographical self-representation in American literature. Most recently, he has focused on computer vision, computational hermeneutics, and the history of machine learning and artificial intelligence.
He is the author of three books: *Modernity and Autobiography in Nineteenth-Century America* (Palgrave, 2017), *Critical Digital Humanities* (University of Illinois Press, 2019), and *The Birth of Computer Vision* (University of Minnesota Press, 2023). He is also co-author, with Rena J. Mosteirin, of the critical code study and poetic interpretation of the Apollo 11 Guidance Computer titled *Moonbit* (punctum books, 2019), and *Perceptron* (punctum books, 2025), a creative and critical account of the perceptron—one of the earliest and most successful machine learning devices—and its inventor, Frank Rosenblatt.

context is manufactured. In implementing memory features, chatbots and GenAI applications, this essay argues, depart from the parallel modes of access to records found in early digital media. These applications work in service of industrializing personal data while opaquely making these objects available for further information extraction and monetization.

Introduction

In early 2025 there was a mostly unremarked upon change in the operation of the large, commercial GenAI-powered chatbots. This change involved the addition of new functionality to chatbot applications. Although initially subtle and largely opaque to users, this feature represents one of the most important recent changes in the everyday experience of commercial GenAI applications. This new feature is now widely known as "memory." Among the major foundation model and chatbot application vendors, this new memory feature was first implemented by OpenAI for their ChatGPT application. Memory is not memory in the cognitive sense (although several contemporary applications borrow concepts from psychology) but is the industrial management of input context for an instruction fine-tuned LLM. Memory, in short, involves the stacking together of selected prior conversations along with the present prompt as input to the LLM. These prior conversations afford new functions and meanings for both application vendors and users: extracting a record of past conversations gives the feeling of personalization—the model will "remember" personal information, past activities, and preferences and these will inform future responses—and most importantly, the ability to shape the "personality" of the chatbot and the simultaneous construction of a user profile work hand in hand to further the impression of continuous learning and autonomous artificial intelligence, the fantasy goal shared by many users and vendors. Through the industrial management of memory-as-context, chatbot applications implementing this memory feature blur the distinction between in-context learning and in-weights learning (Qingxiu Dong et al., 2024; Brown et al., 2020). By continually retrieving and reinserting prior interactions into new contexts, these applications create the appearance of ongoing adaptation and persistent personal memory, even though the underlying language model remains unchanged. This manufactured continuity encourages the perception of a chatbot personality, intensifies anthropomorphic interpretations of model behavior, and

enables the construction of increasingly detailed autobiographical user profiles that can become objects of information extraction and monetization.

GenAI-based chatbots once worked differently. In their initial encounters with these "AI" chatbots, in the first few years since the widespread availability of ChatGPT in November of 2022, many users were confused that they were unable to alter the responses from the model. Some users of ChatGPT were familiar with other Internet-based chatbots that worked differently. Take, for example, Microsoft's extremely short-lived "Tay" bot. Tay was an experimental chatbot released in March 2016 and interacted with via Twitter. While Tay was an adaptive conversational chatbot system and it was based on a language model, its architecture was based on different technologies than the Transformer-based LLMs powering contemporary GenAI chatbots. After running for barely twenty-four hours, the Tay project was canceled due to its alteration by malicious users who prompted the chatbot to engage in far right-wing conspiracies and hate speech (Wolf et al., 2017; Lee, 2016). Tay's responses and tone could be modified through interaction; the users of new GenAI-based chatbots, on the other hand, quickly learned that they were not able to modify the behavior of these chatbots beyond the present chat. Each new chat session, in an LLM-powered application like ChatGPT, resulted in what appeared to be a fresh start.

Using ChatGPT and experiencing the lack of persistent memory provided users with a learning opportunity to understand how LLMs are trained and how they function as inference engines. Those users, especially academic users, who were prompting chatbots to elicit biographical information about themselves, in this moment prior to the addition of an automated web search, were dismayed to see obsolete information or, even more likely, hallucinated outputs. Responding with the correct information, some might have prompted the model again and felt momentarily satisfied in seeing the response reflect these corrections or newly supplied information—until they tried their query again in a new session or asked others to report what they had seen as a response. Basic contemporary AI literacy developed in the past few years has focused on demystifying the long and costly training period during which various internal network architectural components including billions and now trillions of parameters are updated and the later-stage fine-tuning process through which models are modified for use in chatbots. Users of chatbots learned that after these training procedures, models are fixed and frozen. They are placed into a read-only mode. This moment provided some

important clarifications about what LLMs were and were not. They were widely understood as poor replacements for knowledge bases and web search tools. They were also understood as limited by their training data and provided context. Users developed workarounds and techniques: they experimented with prompts that altered the responses to fit specific needs for specific tasks. They shared these with others so that these customized instructions could be imported and pasted into the chatbot input box. The development, in early 2025, of a memory feature changed the nature of interactions with models. Now one no longer needed to archive past interactions and custom prompts. The cutting and pasting of one's own prior chats is no longer necessary: the chatbot already has access to user preferences, special behaviors, past chats, and personal information. It may even give the impression that it possesses a persistent "personality."

In present use, context names the collective set of input sequences provided to an autoregressive language model. Due to autoregression, contexts are iteratively assembled; each output becomes input until a stopping condition is reached. The typical context for a chatbot contains a number of distinct textual fragments. These include system prompts, private instructions that guide the chatbot's application provided by the application vendor, the present history of turn-taking conversational pairings between the user and the chatbot assistant, along with the current prompt. In extending context to include extractions from stored prior chat sessions, GenAI platform vendors have enabled the impression of state into what is a stateless model. Like other inputs that comprise the context, the contents and procedures used to extract "memory" fragments are opaque to end-users. That LLMs are stateless is now hard for everyday users to understand. During a session, the conversational chatbot learns from the context provided but models are never modified. The behind-the-scenes manipulation of context enables many of the recent developments from web search to retrieval augmented generation (RAG). The memory feature is now standard among chatbot applications and more importantly, along with tool calling functions, it is the backbone that powers the so-called agentic AI applications like Claude Cowork, OpenAI Codex, and the open-source OpenClaw toolkit that I will shortly be discussing.

As suggested above, a key aspect of contemporary LLMs is that all inputs are treated as text (Weatherby, 2025). While it is true that all inputs are tokenized prior to inference, the entire context is assembled from textual documents.

Mercedes Bunz analyzes generated or what she terms calculated language as the product of intertextuality due to the nature of the modeled sources and probabilistic language modeling (Bunz, 2026). We might add to Bunz's analysis the intertextual nature of context management in higher level chatbot-based applications; the automatic assemblage of context increasingly blends fragments of relevant texts along with user-supplied input. As GenAI technologies increasingly become components of cultural and ideological infrastructure, understanding the ways in which sources are identified, extracted, and blended as the context becomes critical.[2] Even if objects are retrieved using other mechanisms such as queries from a vector database, the objects in the database are proxies and eventually indices for the text that will need to be extracted and inserted into the context. Because LLMs are autoregressive, the entire input is carried forward and extended by the next generated token at each iteration of generation. This means that platform-supplied instructions that take the form of system prompts are text fragments.[3] While these text instructions are sometimes marked up with special instructions, they are text inputs that preface the body of the text that follows. Other material to be processed by the model, such as the transcript of preceding dialogue and the content of stored memories, also enters the tokenizer as a stream of textual input. While multimodal models can model different categories of input objects, and images are "tokenized" by vision transformers in a manner not so dissimilar from language (Dosovitskiy et al., 2020), these other modalities are handled by different subnetwork architectures at present.

There is a crucial distinction to be made between the sort of "memory" that I am discussing in this essay and the notion that LLMs are providing access to cultural memory. There are also important conversations about the degree to which LLMs and other machine learning models encode information about the world that can be retrieved and reproduced. In this context, researchers would examine these models for their ability to both generalize, which is to say produce

[2] The degree to which LLMs and GenAI applications are cultural technologies depends a great deal on the understanding of what has been assembled as context and the relation between a model's baseline response and the response informed by this context. On AI as a "cultural technology," see Farrell (2025) and Underwood (2025).

[3] M. Beatrice Fazi has analyzed development of automated "alignment" (extending the RLHF paradigm) in the context of "constitutional AI" as the next stage in efforts to install "human-centered" guardrails and guide model responses (Fazi, 2026).

decision criteria that would enable models to address new, unseen data, and to memorize, which would be the exact reproduction of input data as output.[4] These questions have concerned many critical AI researchers, but this kind of memory is not the topic of this essay. A useful analogy can be found in the technical distinction between what is termed "in-weights" and "in-context" learning. The distinction between in-weights and in-context learning is critical to understanding the operation of LLMs. Learning from context is what happens during inference; examples, references, and other inputs can be provided as part of the context along with a user prompt. The former concerns what has been learned from training data and the latter concerns what can be learned from potentially novel data presented as input.[5] In-context is a novel form of learning that has been enabled by the Transformer neural network architecture; it is the multi-head self-attention mechanism that makes learning from context, from long sequences of input data, possible. This attention mechanism is what distinguishes the Transformer architecture from earlier architectures, such as recurrent and convolutional neural networks.[6] It is what has diminished the problem of polysemy in language modeling and what has enabled large language models to address language and not simply tokens. The memory of concern to this essay is thus entirely a product of in-context management and learning, for this learning takes place in every inference. As discussed above, this is in-context, not in-weights, learning and this learning does not modify any model parameters. While records of chat dialogues, depending upon selected parameters or decisions made by application vendors and platforms, may be used as training data for future models, the use of these dialogues would not directly update the model. Because chatbots are designed to recursively supply previous prompts and outputs as inputs, they are continuously performing in-context learning. The appearance of memory of the kind that I am addressing now is thus illusory, held for a moment in an assembled context window. This illusory memory, drawn from formatted

[4] The degree to which we can call stored inputs "memory" in large language models and neural networks in general is contested. This has been the case since the invention of these networks in the mid-to-late 1950s (Dobson, 2023). On memorization in neural networks and LLMs more specifically, see Tirumala et al. (2022) and Chang et al. (2023).

[5] There is much more to be said about the roles that GenAI applications might play in enacting cultural or collective memory from their training data and as a tool that is increasingly shaping the production of new cultural objects. On this topic, see Schuh (2024) and Manovich (2025).

[6] On the need for formal analysis of LLMs and their outputs to include an analysis of the particular neural network architecture used, see Dobson (2025).

textual fragments punctuated with special tokens and encoded instructions from users and application vendors, is the target of today's new industrialization of memory.

Digital Industrialized Memory

The digital media theorist and philosopher Bernard Stiegler conceptualized a transformation of memory that he termed industrial through his analysis of digital technologies, especially the new devices and media appearing in the early twenty-first century. Stiegler argues that the development of analog mass communication systems extended not only the distance between viewers and broadcasters but also the available modes of interaction with media. Stiegler describes twentieth-century cultural technologies as industrialized due to the lack of parallel access to the objects of inscription and distribution. He writes: "One cannot be a reader without being able to write; in the case of analog recording, one can—and typically does—receive audiovisual messages without having the ability to produce them oneself. Thus, industrialization—defined as the separation of producers and consumers—comes into being" (Stiegler, 2010: 77). This division of access is at the core of Stiegler's argument about the industrialization of media. It is also what he saw as being disrupted by Internet technologies in a hopeful early twenty-first-century moment. These "participative technologies *that no longer impose the producer/consumer opposition*" (Stiegler, 2010: 83) were signs, to Stiegler, of an emergent epochal shift. On the early Internet, any reader could also be a writer; YouTube is a platform for uploading videos and for viewing content.

Today's generative AI applications represent a regression from the parallel modes of access to inscription found in prior media technologies. The LLMs powering contemporary systems are large, resource-intensive, and closely guarded proprietary objects. As mentioned above, users of chatbot applications have no possibility of correcting or updating the model itself with new knowledge—they remain in a consumer relation to LLMs. With the incorporation of the new memory features, there has been a fundamental transformation in the presentation of generative AI applications; while the models strictly implement industrialized modes of access, the applications present themselves as if they do not. Storing, summarizing, and retrieving past conversations for future generative activity makes users feel more like producers. These users are invited to imagine

that they are modifying the models and the knowledge stored within them. As memory archives increase, chatbot responses become more personalized, as generated responses are pushed by the weight of these past conversations into smaller and more restricted spaces.

GenAI applications are bound by conditions that make them a dubious source of recall. Sharing individual responses to a prompt, to demonstrate what "AI" had to say about this question or text, provides little insight to others, for this response is not likely to be exactly reproducible or verifiable. Fabian Offert and Ranjodh Singh Dhaliwal point to this almost ubiquitous problem in humanistic Critical AI studies by opening their polemical essay on the emergent field with the question "When is Generative AI 'wrong'?" (Offert and Dhaliwal, 2024). That is a feature of stochastic generation and increasingly a behavior compounded by the automatic insertion of content retrieved from the web and memory into the context. We might say that generative AI applications have an imperfect memory by design but nonetheless they have the potential to serve as useful recording devices, especially in relation to iterative and interactive use with humans. As users build archives of past interactions and these are retained as context, GenAI applications are increasingly becoming individualized technologies of the self. I borrow Stiegler's appropriation of hypomnemata from Michel Foucault to formulate an analysis of the memory fragments created by applications that depend upon memory-as-context. Digital technology was not introduced into Foucault's thinking by Stiegler, for it was already there in his theorization of hypomnemata. While articulating his account of writing the self, Foucault was thinking about technology, specifically about the ways in which writing technologies function as aids to memory and collect the material by which one constitutes oneself. The copybook, the original hypomnemata, as Foucault says in an interview in Berkeley in 1983 "was coming into vogue at Plato's time for personal and administrative use" (Foucault, 2010: 363). "This new technology," Foucault continues, "was as disrupting as the introduction of the computer into private life today. It seems to me the question of writing and the self must be posed in terms of the technical and material framework in which it arose." Pressed by his interlocutors to give more examples of the contents of these hypomnemata, Foucault says of these books, "Into them one entered quotations, fragments of works, examples, and actions to which one had been witness or of which one had read the account, reflections or reasons which one had heard or

which had come to mind. They constituted a material memory of things read, heard, or thought" (Foucault, 2010: 364). While primarily administrative in nature, these copybooks were also used in personal life, as a technology of the self. They were not, however, used in service of giving "an account of oneself," as in later diaries and confessional Christian autobiography (Foucault, 2010: 364). The computer, as Foucault notes, functions as an artificial memory aid and provides access to the materials of self-making. Technology certainly expands the scale of storage and retrieval of archival documents over prior memory technologies. The fragmentary and encyclopedic nature of Foucault's copybook seems an apt description of memory fragments produced by contemporary GenAI applications from user interactions. How these fragments are used, how they might aid in the constitution of the self, remains less clear. Asymmetric modes of access to models, opacity in the automatic memories produced, and the inability to determine which memories are used and how outputs were produced from these selected memories present a distinct departure from prior technologies.

Creating Memory as Context

Prior to the commercial appearance of memory features, academic researchers had developed early memory technologies by using open-weights models and creating third-party platforms for using foundation models. The "MemoryBank" application developed by Wanjun Zhong et al. in 2024 proposed both the pipeline for managing memory and the theoretical framework and language for describing how memory might work in these systems (Zhong et al., 2024). MemoryBank is an application implemented using some modifications on top of a standard RAG pipeline in which fragments of text—the material labeled by the application as memories—and the present prompt are encoded as embeddings with a Transformer-based model and then retrieved with a similarity query of that modeled embedding space. Only the most relevant memories are retrieved for pushing into the context. The encoded "memories" in MemoryBank include direct fragments of past chats, event summaries over a range of past chats (e.g., daily) produced with prompting from the LLM, and LLM-generated "personality traits" of the user. While encoded for efficient retrieval, such a pipeline also requires storing the encoded texts as plain text in order to insert them into the context. Zhong et al. conceive of the MemoryBank as a "long-term

memory module" (Zhong et al., 2024: 19727) that adds new affordances to an unmodified chatbot implemented with an instruction fine-tuned LLM: "MemoryBank enhances the ability to maintain context over time, recall relevant information, and understand user personality. The memory updating mechanism of MemoryBank improves the anthropomorphism of AI in long-term interactions scenarios" (Zhong et al., 2024: 19730). This innovation produces strategies for managing context within an environment that seeks to encourage continual use. That continual use would otherwise result in an ever-growing token count as the context size increases. While context windows—the number of input tokens handled by a model—have been increasing, there are many inefficiencies in preserving a large context. We can think of the management strategy introduced with MemoryBank as directed toward, and optimizing for, memory-as-context.

A key question for our analysis concerns the audience of these generated memories. Who, we might ask, is this memory for? Under the pretense of personalization, LLM-enabled memory is frequently presented as for the user: to make their life easier, to render their results local and more specific, to reduce the amount of context that they have to provide with their queries and instructions. Yet what are preserved as memories are undeniably also functioning for the benefit of the application and platform. The utility of LLMs and chatbots was initially limited by their statelessness. Without persistent storage, large contexts had to be appended to user prompts. The larger the context, the higher the inference cost. Each new token adds an immense number of matrix multiplications. Traditional context management pipelines, such as RAG, add a great deal of complexity and have the potential for using large amounts of storage space. There is, in short, as much a need for "memory" on the part of the application as there is for the user. MemoryBank is infrastructural software. It applies familiar concepts from information retrieval systems—embedding, storage, and retrieval—to chatbots by using the LLM as a compression mechanism to both summarize and embed text and to generate from a context constructed from relevant compressed fragments. This compression is performed in service of a reduction of the context provided as input. An efficient search and retrieval mechanism also makes it possible to preserve and extract information from these memories.

In their user-facing documentation, OpenAI describes their memory feature for ChatGPT in the following terms:

> ChatGPT can remember useful details between chats, making its responses more personalized and relevant. As you chat with ChatGPT, whether you're typing, talking, or asking it to generate an image, it can remember helpful context from earlier conversations, such as your preferences and interests, and use that to tailor its responses. (OpenAI, 2026).

The recursive aspect of memory production, in all contemporary applications of this technology, renders all memories synthetic. Memories are produced as text with the assistance of the language model. Whether summarized or retrieved in whole, the identification and extraction of prior conversations take place through language modeling. The details of this operation are generally withheld from chatbot users; while the implementation of the memory feature is generally an opaque vendor-specific mechanism, a case study of an open-source implementation of an agentic AI application provides some insight into contemporary approaches, functioning as an analogue, if not a direct comparison.

OpenAI's emphasis on memory as enabling personalization in ChatGPT suggests the automatic construction of a user profile from these memories, although in terms less explicit than the description found in the MemoryBank work. In their article, Zhong et al. invoke the creation of a user profile from past chat sessions as enabling a "detailed user portrait" (Zhong et al., 2024: 19730) that will promote "a more personalized interaction" (Zhong et al., 2024: 19724) between the chatbot assistant and user. The two prompts they provide to be used alongside prior chat pairings aim to extract unstructured and ill-defined traits from prior user prompts:

> "Based on the following dialogue, please summarize the user's personality traits and emotions. [dialog]" or "The following are the user's exhibited personality traits and emotions throughout multiple days. Please provide a highly concise and general summary of the user's personality [daily Personalities]." (Zhong et al., 2024: 19725)

The sort of personalization involved in summarizing chatbot interactions is distinct from the profile construction used in other machine learning contexts.

In those contexts, a profile might be drawn from coded behaviors and take the form of what Antoinette Rouvroy, in her explication of the theory of algorithmic governmentality, theorizes as a data double (Rouvroy, 2013). Unlike the probabilistic data double of algorithmic governmentality, the memory-generated user profile is directly informed by interactions, here, patterns of language use in prior prompts, from and with the user of a chatbot. Statistical profiles of the sort analyzed by Rouvroy are predictive; they function as descriptions of likely attributes or actions but are not traceable back to specific actions of users. Such profiles are data doubles, according to Rouvroy, because they are not interpretable in terms of the individual but rather in terms of a composite surrogate that is likely to be used to make decisions about multiple individuals. Profiles are also not likely to function as ideal subjects, in the sense used by Olga Goriunova to analyze what she calls the "abstract people of AI" (Goriunova, 2025). The profile constructed by such systems is not necessarily compared to others or used to discipline or correct the user—although one could imagine that this might change. Memory, in GenAI applications, may introduce generalizations about individuals through the summarization of interactions, but at this point the "profiles," insofar as we might describe these summarizations as profiles, are iteratively reconstructed. The generated responses to the prompts used by Zhong et al. are likely to be model-specific and dependent upon generation parameters, resulting in different descriptive categories or labels for traits and emotions on different iterations applied to the same dialogues.

Agentic AI and OpenClaw

The term "agentic AI" was appropriated from prior technologies to give coherence to the patchwork of instruction fine-tuned language models, tool calling functions, and the context management functions that make up contemporary LLM-based memory systems. Exploiting the ELIZA-effect inspiring anthropomorphic behavior (Ciston et al., 2026) trained into instruction fine-tuned models, one might say that agentic AI applications are powered by dreams. Taking up the open-source agentic system known as OpenClaw provides a useful case study for the analysis of contemporary industrial techniques to handle the creation and management of memory. The OpenClaw developers describe the application as "The AI that really does things" and suggest to potential users an array of possible uses. OpenClaw, the developers write,

"organizes your inbox, sends emails, manages your calendar, checks you in for flights. All from WhatsApp, Telegram, or any chat app you already use." These two modes of interaction—driving applications and responding to user prompts—are the major differentiators for these agentic applications. Unlike a hosted application, OpenClaw can be installed on a user's home computer and can be run totally offline using a local LLM running with the Ollama application or remotely via API calls. The release of the package drove sales of Apple's Mac mini. This is a small form-factor, always-on computer with shared memory between the system and co-processors used for accelerating the operations required for neural networks that is ideal for local inference of smaller LLMs. In using a local application on local hardware and leveraging a local model, users have some assurance that their chats and produced objects and the memory of their past interactions will stay tethered to their devices.

My casting of OpenClaw's management of memory-as-context as a mode of industrialization is intended to draw attention to a certain amount of opacity and one-sidedness in its construction and revision of these memory fragments. That OpenClaw is open source and that it uses markdown, a human-readable and -writable plain-text format, for storing these memories does not negate the industrial quality of its memory management. While the documents it produces and uses are human-readable and the prompts used to create these documents are transparently provided in the application's TypeScript source code, these processes are handed off to an LLM, typically but not always an API-based call to a foundation model. That black-box nature of such strategies for creating and managing memory-as-context means that while synthesized memories might be editable, the ongoing pipeline of creating and revising these memories is handled autonomously and with a significant degree of opacity.[7] The formation of memories and their continual automatic revision happen with minimal user input, and any human input is easily overwritten by newly constructed memories. OpenClaw, like any application built on top of an LLM, also preserves the asymmetric producer/consumer relation because this relationship is an inherent attribute of the division between model training and inference.

The instructions used to compact prior conversations are structured around the summarization and preservation of past interactions and responses.

[7] An excellent critique of the black box metaphor in LLM-based systems can be found in Nia Judelson and Maggie Dryden (2024).

In many memory documents, we find in the content of the memory an instruction-oriented record for past interactions. These records are not direct copies but summarized or, in the language of OpenClaw, compacted chats. The compaction process is automated in OpenClaw and is executed when the context size approaches a threshold limit. It can also be executed on demand by a user interacting with OpenClaw. While the OpenClaw code presents some outputs as actions and refers to the application as an agent, I'd like to bracket the question of agency in my analysis of the memory function of contemporary LLM-based applications. To be sure, there are records of executions in the form of tool calling and output from these executions. In naming these function calls "actions," much is made unclear.

The following instructions are used as a prompt to compact or summarize the current context window used by OpenClaw:

> Merge these partial summaries into a single cohesive summary.
> MUST PRESERVE:
> - Active tasks and their current status (in-progress, blocked, pending)
> - Batch operation progress (e.g., "5/17 items completed")
> - The last thing the user requested and what was being done about it
> - Decisions made and their rationale
> - TODOs, open questions, and constraints
> - Any commitments or follow-ups promised
> PRIORITIZE recent context over older history. The agent needs to know what it was doing, not just what was discussed. (OpenClaw, 2026a)

As already mentioned, in OpenClaw memories are produced through the compaction mechanism. This process involves writing context to a human-readable markdown file for future use. By "compacting" context into short, summarized descriptions, the context can be reduced in size but more importantly, persisted to file. As a persistent file, on a storage device, these memories are thus able to be reloaded into the present context after a restart of OpenClaw or the entire system. Like the use of memory in a typical commercial chatbot application, this managed memory-as-context gives the sense of continuity to interactions with OpenClaw. Reloading the managed context would ideally be no different from a continuous chat dialogue. The use of compaction to summarize past interactions, however, may introduce some loss of context, call it

forgetfulness, in either the dropping of useful or important tokens or incorrect summarization. We might even analyze the presence of hallucinated compacted memories as the introduction of false memories into the managed memory.

There are several distinct categories of persistent memory storage used by the OpenClaw package. We could think of these memory documents as belonging to different genres, being written in different voices, and directed toward different audiences. The question of audience for the various documents included in the assembled context for a chatbot is one confronted by anyone examining a commercial application system prompt or attempting to parse Claude's "constitution." That latter document, as Anthropic explains in their introduction, might confuse readers: "The document is written with Claude as its primary audience, so it might read differently than you'd expect. For example, it's optimized for precision over accessibility, and it covers various topics that may be of less interest to human readers" (Askell et al., 2026). One particularly intriguing persistent storage location in OpenClaw has the fascinating title of SOUL.md. This file, this memory, is delivered with a template. It contains boilerplate text that has been designed to be overwritten by OpenClaw once the application is up and running. The document, as provided by the developers to users, outlines a self-revisable system prompt that seeks, via direct address to the model, to insert "depth" by flipping the archetypical system prompt ("You are a helpful assistant") on its head. It begins with a comment: "# SOUL.md - Who You Are." The next line: "You're not a chatbot. You're becoming someone." The document continues to define, under markdown headings, "Core Truths," "Boundaries," "Vibe," "Continuity," and "Related." The "Continuity" heading defines the terms of this document: "Each session, you wake up fresh. These files _are_ your memory. Read them. Update them. They're how you persist. If you change this file, tell the user — it's your soul, and they should know" (OpenClaw, 2026b).

The SOUL.md file is descriptive yet has no proper object. We know, of course, that there is nothing—no features, no neurons, no pattern of activation within the neural network—being identified as the soul of this machine. As a self-revisable and human-editable markdown file, there is no definable chain of provenance that could be used to reconstruct this file. It is a prompt disguised as a memory. This is because everything is context in an LLM. The only way for the contents of SOUL.md to enter the model is as context, tokenized and inferred as

all inputs must be. In taking up the memory metaphor and reading GenAI-produced documents through Foucault's critique of writing technologies, we can conceptualize SOUL.md and compacted user interactions as the materials through which autobiographical self-narratives are constituted—for the human user and the anthropomorphized LLM assistant. These documents, I'm arguing, form a revisable site of retention for the illusion of persistence in chatbots and agentic AI.

Dreaming and Longer-Term Self-Revisable Memory

Agentic applications like OpenClaw and research projects like MemoryBank further the anthropomorphism involved in the memory concept through the invocation of a schema of different memory categories or types including what they term short- and long-term memory. OpenClaw also makes use of additional psychological concepts through the developers' implementation of another memory revision mechanism that they call "dreaming." To dream, for the developers of OpenClaw, means the revision of stored, persistent summarized memories from past interactions into an ongoing higher-level narrative of all of these interactions. If the addition of memory and the management of memory-as-context to chatbots work asymmetrically to serve infrastructural needs in the form of compression and cost reduction under the rationale of personalization, the dreaming function enhances the illusion of the chatbot assistant as possessing an autobiographical self through these same infrastructural needs. This is to say that dreaming manufactures, for the responses provided by the instruction fine-tuned assistant, the appearance of a persistent "self" with an interior life through prompting and iterative scheduling. That these relatively simple strategies preserve the appearance of a roughly similar response across different models suggests a growing convergence in model training techniques.

Dreaming leverages the solution to an *application-side problem of unconstrained growing context found in the development of memory infrastructural systems to further the fantasies of autonomous artificial intelligence.* The higher-level memory organizing prompt reads quite simply: "Write a dream diary entry from these memory fragments" (OpenClaw, 2026c). Dream narratives, according to the logic expressed by the developers in the OpenClaw code, are designed to render more coherent the individual memories compressed from contexts drawn from multiple task executions and chat dialogues. At the present time, the dreaming

function is optional and disabled by default. When enabled, this function implements three dream states: light, deep, and REM. Dreaming, of course, involves no modification of LLM state and takes no longer to execute than the inference time for the supplied inputs. The dreaming prompts make explicit a division that is tacit in the memories—the distinction between the "I" of the chatbot discourse and that of the human user. The instructions for constructing a dream diary are invoked as a series of directives combined into the NARRATIVE_SYSTEM_PROMPT variable and define the "voice" to be used in the summarization of the fragmented memories as singular and human-like. It begins with "You are keeping a dream diary. Write a single entry in first person" and continues to define the voice and tone for the response. "Write like a poet," the programmers instruct the model, "who happens to be a programmer — sensory, warm, occasionally funny." A series of succinct rules follows:

- Draw from the memory fragments provided — weave them into the entry.
- Never say "I'm dreaming", "in my dream", "as I dream", or any meta-commentary about dreaming.
- Never mention "AI", "agent", "LLM", "model", "language model", or any technical self-reference.
- Do NOT use markdown headers, bullet points, or any formatting — just flowing prose.
- Keep it between 80-180 words. Quality over quantity.
- Output ONLY the diary entry. No preamble, no sign-off, no commentary. (OpenClaw, 2026c)

These instructions collectively attempt to shift the generation of outputs from instruction fine-tuned defaults that tend to be direct, precise, formatted, and engaged in some disavowal of agency. There is an explicit instruction to avoid the common introductory stock phrase of "As an AI language model." These define the dream diary as a short but creative record of past agent-user interactions, written in first-person. The diary compresses and rewrites summarized memories produced from the turn-based dialogues outputted by instruction fine-tuned LLMs, again containing both user-authored prompts and generated output, into what is presented as the "memory" of the agent of these interactions. With the addition of the dreaming state, OpenClaw implements a multistage rewriting workflow that functions to manage supplied context through the continuous

extraction of descriptive text from interactions with chatbots. Rewriting of text turns out to be a common use of LLMs; in research applications for text analysis, rewriting provides a different way of saying something that might be more suitable to information extraction purposes, whether using traditional natural language processing or more advanced techniques. This iterative rewriting both reproduces, at a formal level, the stochasticity of generative models and leverages it, drawing upon the slipperiness of the signifier to reencode language. Critical analysis of the dreaming function in OpenClaw foregrounds the automatic, semi-autonomous, and continual revision of these autobiographical-like self-narratives that function primarily to forward the illusion of interior life. The constitution of OpenClaw's assistant can thus be conceptualized as a guided intertextual blending of hypomnemata-like extracted records. The SOUL.md and dream diaries vividly illustrate the asymmetries of industrialized memory production in GenAI applications as they are scratchpads of memory that operate primarily for the application and platform and only secondarily for the user, in reinforcing fantasies of the "self" created by agentic AI applications.

Conclusion

Because of its slipperiness as a signifier, the overly broad ways in which it is used, and the nature of its technical details, which are frequently complex and opaque, it is especially productive to cast the critique of "artificial intelligence" through two different readings of the assemblage. Building on Manuel DeLanda's expanded assemblage theory, Simon Lindgren suggests that we include as objects of analysis "not only technology (its tools and machines) but also: *humans* the behaviors of whom form both the input and output of AI systems; and *material and symbolic artifacts*—patterns of community, organization, design, and architecture that are affected by AI systems" (Lindgren, 2024: 26). Familiar, perhaps, from science and technology studies, Lindgren's approach to AI questions takes up a wide and diverse set of actors and objects that interact with these technologies. This conceptualization of the assemblage, importantly, enables us to examine AI imaginaries, the ways in which we talk about AI, the fears, dreams, and fantasies attached to these technologies, alongside other critical objects of analysis such as the technologies, data, and people who produce these systems. Google's "AI Overviews" and other features added to existing applications from Adobe's Photoshop to Microsoft's PowerPoint, for example, are

clearly embedded within already-existing commercial technological systems and are thus poised to take advantage of users' familiarity with these applications and trust in these interfaces and their outputs.

The other use of assemblage is more technical and reframes generative AI applications in relation to the modules, components, and other part-objects that are brought together to make them work. Tracing the technical assemblage involves understanding each element in relation to the others. Thinking in this way foregrounds the flow of data, through both the entire assemblage and individual components. It means a focus on the inputs and outputs in this data flow. In some contexts, we might call this assemblage a pipeline. Pipelines can be abstractions, but they are also material in that they are the structures required to build applications. This technical understanding of the assemblage is especially useful in the critique of autonomy that so frequently accompanies AI discourse. Autonomy, in this context, is the sense that when using a generative AI-based application like ChatGPT, users are interacting directly with a model and that all the features and functionality are intrinsic to the model itself and not the heavily managed context. This is to say the fantasy of immanent knowledge, the desire for an artificial intelligence lying dormant within the neural network, waiting to be called into being by the prompting of users.

It is increasingly hard to hold any assumptions about the pipeline and generation process used in chatbot applications. While critics like N. Katherine Hayles have suggested that it might be productive to read outputs in relation to texts found in training data, contemporary training procedures, system prompts, and the manufactured context of memory-enabled chatbots deeply complicate any causal or even influential link between documents used for pre-training and outputs.[8] As opaque units in an exploding number of pipelines, LLMs are troublesome context manufacturers. When the iteration and recursion built into autoregression are applied in the production of derived context—the summarization, synthesis, and filtering involved in what I have described as *context-as-memory*—provenance data are lost and citation and source are delinked, even as surveillant and extractive archives are multiplied. Critical AI studies of contemporary "AI" that attempt to understand how meaning is made with

[8] Writing of GPT-3, an early-generation LLM, Hayles argues that readers might speculate on the "source texts" used as training data by reading and interpreting outputs to prompts that invoke known texts or referents (Hayles, 2022).

generative LLM-based chatbot applications while neglecting the larger assemblage, the context in which these technologies function, cannot make supportable claims about the link between training data and outputs if the structuring of input data is neglected. The critical reading strategy of memory systems offered in this essay shares much with the systems-level analysis proposed by Tyler Shoemaker (Shoemaker, 2026). Shoemaker argues that humanists directing critical questions toward GenAI should read the system rather than the model, including the systems that produce the model. MemoryBank and OpenClaw provide accessible, open, and documented implementations of memory-as-context management systems and agentic AI platforms. Taking up these systems and solutions as case studies enables a much closer systems-level analysis of several components involved in the contemporary AI assemblage than would be possible with closed-source and platform-based systems.

With the development of personalization and persistent automatically summarized context management features, chatbots and agentic applications have moved into the mode of industrial memory management. While open-source applications may expose a small number of sites for read-write activity, enabling users to selectively edit, insert, and delete the store of memories associated with their activity, the sheer volume of tokens generated and the mechanisms through which these applications consolidate past activities deskill and deauthorize their users. Memory-as-context furthers the personalization and individualization of model responses and risks amplifying already existing tendencies toward sycophancy, in terms of model response, and confirmation bias, on the part of the user. If the digital once promised a loosening up of the formerly strictly defined producer and consumer roles and relations, the advent of corporatized platform-based endlessly recursive writing machines has terminated that once hopeful dream. At the same time, we might ask what possibilities might arise in the more transparent and user-driven creation of memories from interactions with chatbots and other LLM-powered applications. What would it mean to truly have an *aide-mémoire* in persisted, archived interactions with a chatbot? How would such improved hypomnemata inform, for better or worse, self-making and self-governing activities? While earlier forms of digital media promised parallel access to distribution mechanisms that rendered them more participatory in nature—readers could easily become

writers themselves—memory-enabled chatbot applications automate inscription. Chatbot application users still produce text, but increasingly lose authority and control over its preservation, revision, and interpretation. Their records, their autobiographical traces of their interactions with chatbots, become recursively rewritten by platform infrastructure and deployed on their behalf, frequently without their knowledge or consent.

References

Askell, Amanda, et al. (2026), 'Claude's Constitution', January 21. Available at: https://www-cdn.anthropic.com/d0636f72a9493d279ed36b33987da3430bcb5911/claudes-constitution_webPDF_26-02.02a.pdf

Brown, Tom B., et al. (2020) 'Language Models Are Few-Shot Learners', paper presented at 34th Conference on Neural Information Processing Systems (NeurIPS 2020).

Bunz, Mercedes (2026) 'On the Calculation of Meaning, or Doing Words without Things', *Critical Inquiry*, 52(3), pp. 423–446. Available at: https://doi.org/10.1086/739755.

Chang, Kent K., Cramer, Mackenzie, Soni, Sandeep and Bamman, David (2023) 'Speak, Memory: An Archaeology of Books Known to ChatGPT/GPT-4', arXiv preprint, arXiv:2305.00118. Available at: http://arxiv.org/abs/2305.00118.

Ciston, Sarah, Berry, David M., Hay, Anthony C., Marino, Mark C., Millican, Peter, Shrager, Jeff, Schwarz, Arthur I. and Weil, Peggy (2026) *Inventing ELIZA: How the First Chatbot Shaped the Future of AI*. Cambridge: MIT Press.

Dobson, James E. (2023) 'Memorization and Memory Devices in Early Machine Learning', *Interfaces*, 4.

Dobson, James E. (2025) 'Beyond Computational Formalism or, Architecture Matters', *Journal of Cultural Analytics*, 10(3). Available at: https://doi.org/10.22148/001c.133923.

Dong, Qingxiu, Li, Lei, Dai, Damai et al. (2024) 'A Survey on In-Context Learning', *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1107–1128. Available at: https://doi.org/10.18653/v1/2024.emnlp-main.64.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, et al. (2020) 'An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale.' Paper presented at International Conference on Learning Representations. Available at: http://arxiv.org/abs/2010.11929.

Farrell, Henry, Gopnik, Alison, Shalizi, Cosma and Evans, James (2025) 'Large AI Models Are Cultural and Social Technologies', *Science*, 387(6739), pp. 1153–1156. Available at: https://doi.org/10.1126/science.adt9819.

Fazi, M. Beatrice (2026) 'Off-Centre AI: On Alignment, Antihumanism and AI Ethics', *Ars & Humanitas*, 20(1), pp. 127–140. Available at: https://doi.org/10.4312/ars.20.1.127-140.

Foucault, Michel (2010) "On the Genealogy of Ethics: An Overview of Work in Progress," in Rabinow, Paul (ed.) The Foucault Reader. New York: Vintage.

Goriunova, Olga (2025) *Ideal Subjects: The Abstract People of AI*. Minneapolis: University of Minnesota Press.

Hayles, N. Katherine (2022) 'Inside the Mind of an AI: Materiality and the Crisis of Representation', *New Literary History*, 54(1), pp. 635–666.

Judelson, Nia and Dryden, Maggie (2024) 'Restaging the Black Box: How Metaphor Informs Large Language Models', *Critical AI*, 2(1). Available at: https://doi.org/10.1215/2834703X-11205196.

Lee, Peter (2016) 'Learning from Tay's Introduction', March 25. Available at: https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/

Lindgren, Simon (2024) *Critical Theory of AI*. Hoboken, NJ: Polity Press.

Manovich, Lev (2025) 'Unreliable Past: Constructing Memory with Generative AI', *Leonardo*, 58(5), pp. 477–483.

Offert, Fabian and Dhaliwal, Ranjodh Singh (2024) 'The Method of Critical AI Studies, A Propaedeutic', arXiv preprint, arXiv:2411.18833. Available at: https://doi.org/10.48550/arXiv.2411.18833.

OpenAI (2026) 'Overview', Memory FAQ. Available at: https://help.openai.com/en/articles/8590148-memory-faq

OpenClaw (2026a) 'compaction.ts'. Available at: https://github.com/openclaw/openclaw/blob/main/src/agents/compaction.ts

OpenClaw (2026b) 'SOUL.md'. Available at: https://github.com/openclaw/openclaw/blob/main/docs/reference/templates/SOUL.md

OpenClaw (2026c) 'dreaming-narrative.ts'. Available at: https://github.com/openclaw/openclaw/blob/main/extensions/memory-core/src/dreaming-narrative.ts

Rouvroy, Antoinette (2013) 'The End(s) of Critique: Data Behaviourism Versus Due Process', in Hildebrandt, Mireille and de Vries, Katja (eds.) *Privacy, Due Process, and the Computational Turn: The Philosophy of Law Meets the Philosophy of Technology*. Routledge.

Schuh, Julien (2024) 'AI As Artificial Memory: A Global Reconfiguration of Our Collective Memory Practices?', *Memory Studies Review*, 1(2), pp. 231–255.

Shoemaker, Tyler (2026) 'Systems Programming the Model', *AI & Society*. Available at: https://doi.org/10.1007/s00146-026-02897-y.

Stiegler, Bernard (2010) 'Memory', in Mitchell, W.J.T. and Hansen, Mark (eds.) *Critical Terms for Media Studies*. Chicago: University of Chicago Press.

Tirumala, Kushal, Markosyan, Aram H., Zettlemoyer, Luke and Aghajanyan, Armen (2022) 'Memorization Without Overfitting: Analyzing the Training Dynamics of Large Language Models', arXiv preprint, arXiv:2205.10770. Available at: http://arxiv.org/abs/2205.10770.

Underwood, Ted (2025) 'The Impact of Language Models on the Humanities and Vice Versa', *Nature Computational Science*, 5, pp. 695–697. Available at: https://doi.org/10.1038/s43588-025-00819-4.

Weatherby, Leif (2025) *Language Machines: Cultural AI and the End of Remainder Humanism.* Minneapolis: University of Minnesota Press.

Wolf, Marty J., Miller, Keith, and Grodzinsky, Frances S. (2017) 'Why We Should Have Seen That Coming', *ACM Computers & Society*, 47(3), pp. 54–64.

Zhong, Wanjun et al. (2024) 'MemoryBank: Enhancing Large Language Models with Long-Term Memory', *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17), pp. 19724–19731.